Data-driven Surrogate Models for Wind Turbine Design and Maintenance Applications

by

Rad Haghi
B.Sc., Chamran University, 2005
M.Sc., Kingston University, 2006
M.Sc., Delft University of Technology, 2011

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Mechanical Engineering

Data-driven Surrogate Models for Wind Turbine Design and Maintenance Applications

by

Rad Haghi
B.Sc., Chamran University, 2005
M.Sc., Kingston University, 2006
M.Sc., Delft University of Technology, 2011

Supervisory Committee

Dr. Curran A. Crawford, Supervisor
(Department of Mechanical Engineering, University of Victoria)

Dr. Bradley J. Buckham, Departmental Member
(Department of Mechanical Engineering, University of Victoria)

Dr. Adam H. Monahan, Outside Member
(School of Earth and Ocean Sciences, University of Victoria)

Dr. Derek R. Bingham, Outside Member
(Department of Statistics and Actuarial Science, Simon Fraser University)

ABSTRACT

There is a gap between the current contribution of wind energy to the global electricity generation mix and its potential capacity. This discrepancy underscores the necessity for addressing social, economic, and technical hurdles that are impeding the broader integration and acceptance of wind energy. The research focuses on tackling the modelling challenges in wind energy by employing Surrogate Model (SM) techniques, combining probabilistic methods, machine learning, and simulation technologies.

This dissertation aims to develop SMs capable of mapping wind time series to the power output as well as extreme and fatigue loads on wind turbines. In this dissertation, I try to answer a number of crucial questions: determining the most effective type of SM for this mapping, identifying the optimal sampling method for building these SMs, extending the applicability of the developed SMs with minimal effort, and leveraging publicly available simulation tools and wind turbine models for turbine health assessment. These objectives are essential for improving wind turbine design, operation, and maintenance, enhancing their efficiency and reliability.

Throughout the dissertation, there is an effort to bridge the gap between theoretical research and practical application. The surrogate models developed are presented as a contribution to the integration of theoretical concepts with practical applications in the field of wind turbine design and maintenance. Central to this research is the development of SMs for effectively mapping wind time series to the extreme and fatigue loads experienced by wind turbines. The aim is to find the optimal SM type that balances accuracy with computational feasibility. As the wind turbine faces diverse conditions, I propose adaptable methodologies to optimize the SM performance across various settings. Additionally, I investigate the potential of combining publicly available wind turbine models with probabilistic data-driven models to assess turbine health.

First, a non-intrusive Polynomial Chaos Expansion (PCE) is constructed based on the outputs from the NREL 5MW Blade Element Momentum (BEM) model, demonstrating the convergence of sectional statistics in the results. Subsequently, I utilize the SM to estimate thrust and torque on the rotor and perform a sensitivity analysis of the extreme loads to the number of Monte Carlo Simulation (MCS) in the SM. Transitioning from the PCE realm, I adopt a sequential Machine Learning (ML) method to map wind time series to the Damage Equivalent Load (DEL)

of wind turbine loads. I demonstrate that the developed SM, based on a Temporal Convolutional Network (TCN)-Fully Connected Neural Network (FCNN) architecture, is capable of capturing the wind turbine structural dynamics. It demonstrates adaptability in digesting the upstream wakes and accurately estimating the DEL utilizing Transfer Learning (TL). Moving beyond purely synthetic data, I propose the development of a probabilistic data-driven model, integrating limited wind turbine measurements with synthetic data for wind turbine health assessment purposes. I illustrate that an Approximate Gaussian Process Regression (AGPR) trained on a year's worth of Supervisory Control and Data Acquisition (SCADA) data, combined with simulation outputs from a publicly available wind turbine model, emerges as a promising probabilistic tool for wind turbine health assessment.

Contents

List of Tables

List of Figures

List of Abbreviations

AGPR Approximate Gaussian Process Regression

ANN Artificial Neural Network

BEM Blade Element Momentum

BR blade root

CM Condition Monitoring

CNN Convolutional Neural Network

CVAE Conditional Variational Autoencoders

DEL Damage Equivalent Load

DLC Design Load Case

DT Digital Twin

FAST Fatigue, Aerodynamics, Structures, and Turbulence

FCNN Fully Connected Neural Network

GHG Greenhouse Gasses

GP Gaussian Process

GPC Gaussian Process Classification

GPU Graphics Processing Unit

GPR Gaussian Process Regression

GQ Gaussian Quadrature

HD Hellinger Distance

IESVic Integrated Energy Systems at the University of Victoria

IPCC Intergovernmental Panel on Climate Change

KL Kullback-Leibler

LSTM Long Short-Term Memory

max maximum

MC Monte Carlo

MCS Monte Carlo Simulation

min minimum

ML Machine Learning

MSE Mean Square Error

NREL National Renewable Energy Lab

NSERC Natural Sciences and Engineering Research Council of Canada

NRMSE Normalized Root Mean Square Error

OFP out-of-plane

PCE Polynomial Chaos Expansion

PLL Predictive Log Likelihood

POD Proper Orthogonal Decomposition

QMC Quasi Monte Carlo

QoI Quantity of Interest

RNN Recurrent Neural Network

SCADA Supervisory Control and Data Acquisition

SGD Stochastic Gradient Descent

SGQ Sparse Gaussian Quadrature

SM Surrogate Model

STD standard deviation

TCN Temporal Convolutional Network

TL Transfer Learning

TB tower-base

TT tower-top

UQ Uncertainty Quantification

UW Uniform-Weibull

UU Uniform-Uniform

pRNG pseudo-Random Number Generator

ACKNOWLEDGEMENTS

Seven years ago, when I left my secure job at Siemens Gamesa Renewables in the Netherlands to move to the West Coast, I could not have anticipated what awaited me. This journey, both personal and professional, has spanned a global pandemic – a once-in-a-century event, and many more personal events! Where a Ph.D. journey starts and is supposed to end, is often far from where it ends. I embarked on this path, aiming for Utopia, an ambitious goal indeed. Yet, it did not end up in Dystopia only through the support and help of many.

First and foremost, my heartfelt thanks go to my supervisor, Dr. Curran Crawford. Your belief in me and patience with the project were the cornerstones of this research's success. The freedom you granted in research – partly due to your busy schedule, but also, I believe, out of trust – allowed me the space to think, experiment, make mistakes, and grow. I deeply appreciate our discussions; they have been hopeful, helpful, and constructive, even at the lowest points of this Ph.D. journey. Your provocative questions sparked my curiosity, and treating me as a peer, not just a trainee, created a safe environment for inquiry and growth. Your awareness of the mental struggles inherent in a Ph.D. journey was a constant source of relief. Our chats about everything and nothing have been immensely enjoyable, and I value your mentorship greatly. My gratitude also extends to my advisory committee members, Dr. Buckham, Dr. Monahan, and Dr. Bingham, for their sparse yet incredibly fruitful supervision and discussions. Special thanks to Dr. Vidal, the external examiner, for thoroughly reviewing my work.

A Ph.D. journey can feel like a gruelling, solitary trek. It would have been far lonelier without my colleagues, fellow students, and office mates. A special shoutout to Dr. Markus Sommerfeld – Markus, thank you for being a fantastic office mate, gym buddy, and friend. Our discussions, dinners with your family, Yuka and Hanna, and the fun times shared are cherished memories. Gratitude to Gerard, Heather and Patrick for their insightful reviews of my work, board game nights and walks around the campus; to Mike C. for being the driving force during lunch hours; and to Mike P. for the over 40s wisdom talks. Special thanks to Pauline Shepherd for her administrative support at the IESVic. You kept the IESVic office a welcoming haven during the pandemic. Your wisdom, stories, and advice have been invaluable.

This research would not have been possible without the financial support from the Natural Sciences and Engineering Research Council of Canada and IESVic. Your assistance has been crucial.

Beyond the department, my friends have been an incredible support system. Arash, thank you for all our talks, agreements, disagreements, schema therapy insights, empathy, and for sharing the house at Foul Bay during the first three years of this Ph.D. Golnar thanks for being an incredible friend in Victoria and Vancouver. A shoutout to the "Unicorn Meetup" team – you were my bubble during the pandemic. Gratitude to the owners and trainers at Forge Training; your workouts were vital to my mental health and well-being.

From afar, I must thank Ms. Tahmasebi. Our sessions and conversations have been invaluable in helping me understand myself better, grow personally, and find peace with life. Also, a big thank you to my European friends – Sepehr, Nima, Dena, Pejman, Ali, Turaj and Francesca – for supporting my decision to leave the Netherlands.

Last but certainly not least, my deepest gratitude to my partner and family. Katerina, having two Ph.D. candidates trying to graduate under one roof could go either way, and thankfully, it went splendidly, largely thanks to you. To my sister and her family, Ghazal, your support has been an invaluable source of strength throughout this journey. To my parents, Nasrin and Foad, your unconditional love and support are the most valuable assets in my life. Mum and Dad, your belief in me, your patience with my slowness and perfectionism, and your encouragement to spread my wings, learn, and grow have been the bedrock of my journey. Words can not express my gratitude and appreciation for all you have done.

If you are reading this hoping to see your name and did not find it, I apologize. Your contribution to this journey is greatly appreciated.

Rad Haghi
Nov 2023, Victoria, BC

To Ava and Edvin,
for all the joy they bring to us.

To Κατερίνα,
for being there for me.

To Nasrin and Foad,
for their unconditional love and support.

تا بدانجا رسید دانش من که بدانم همی که نادانم
- ابوشکور بلخی، شاعرسده چهارم هجری

My knowledge has reached a point
that I know, I know nothing.
–Abu-Shakur Balkhi, 10th-century Iranian poet

Chapter 1

Introduction

> *"What giants?" said Sancho Panza.*
> *"Those thou seest there," answered his master, "with the long arms, and some have them nearly two leagues long."*
> *"Look, your worship," said Sancho; "what we see there are not giants but windmills, and what seem to be their arms are the sails that turned by the wind make the millstone go."*
> *"It is easy to see," replied Don Quixote, "that thou art not used to this business of adventures; those are giants; and if thou art afraid, away with thee out of this and betake thyself to prayer while I engage them in fierce and unequal combat."*
>
> – Miguel de Cervantes, Don Quixote

In the contemporary era, humanity confronts unparalleled challenges posed by climate change. This phenomenon, primarily attributed to unsustainable practices in energy utilization, land exploitation, and consumption habits, has increased the emission of greenhouse gases, thereby intensifying global warming [1]. The forthcoming years are projected to witness an escalation of climate-related threats, both to the natural environment and human societies. Manifestations of these threats include more frequent and severe meteorological phenomena, heightened flood risks in coastal regions, and an increase in both food- and water-borne diseases alongside mental health issues [1]. Additionally, the loss of biodiversity and diminishing food production capacities are emerging as critical concerns [1]. These multifaceted implications of climate change have propelled the issue to the forefront of public discourse

and media coverage [2]. This heightened awareness is also reflected in phenomena such as "Climate Anxiety," as identified in [3], and influences patterns in electoral behaviour [4].

The exploration for a solution to climate change reveals that there is no universal remedy. Achieving the objectives of the Paris Agreement necessitates a fundamental transformation in our economic frameworks, energy infrastructures, and lifestyles to curtail Greenhouse Gasses emissions. An analysis of sectoral contributions to global Greenhouse Gasses (GHG) emissions highlights two primary areas of concern: transportation and electricity generation (Figure 1.1). A significant reduction in the impact of the transportation sector can be realized through the electrification of transport systems, notably the adoption of electric vehicles. However, this raises a pertinent question: what is the source of the electricity for these vehicles? A dual strategy is essential: promoting electrification while simultaneously reducing emissions from electricity generation systems. The Intergovernmental Panel on Climate Change (IPCC) report [1] suggests that renewable energy sources could potentially replace fossil fuel-based electricity generation, thereby diminishing GHG emissions from this sector.

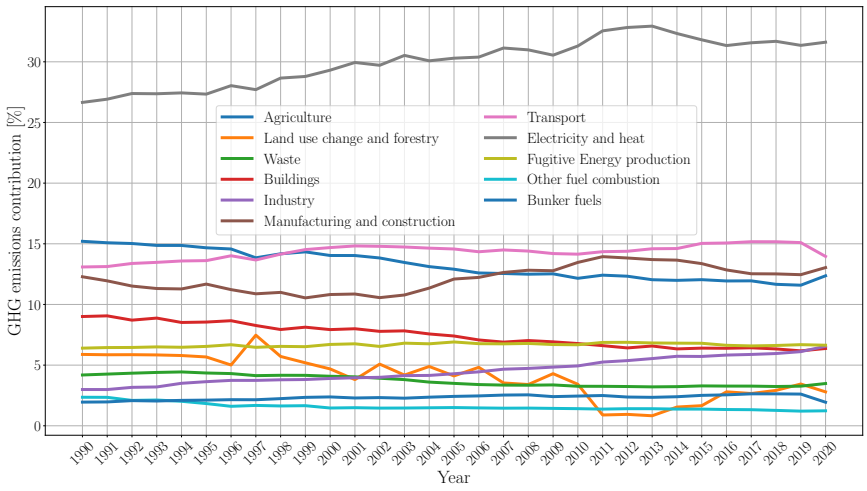


Figure 1.1: GHG emission contributions from different sectors from 1990 to 2020 [5]

The transition from GHG sources of electricity generation to renewable energy is underway, with wind energy experiencing notable growth. From the generation of approximately 4 TWh in 1990, global electricity production from wind energy surged to 2140 TWh by 2022. This represents a substantial increase in power generation from

this renewable source. Despite this progress, wind energy accounted for only 7.5% of the total global electricity generation mix in 2022, while coal still contributed an important 35% [6] (Figure 1.2). The difference between the current wind energy contribution to electricity generation and its potential raises important questions about what is blocking further growth.

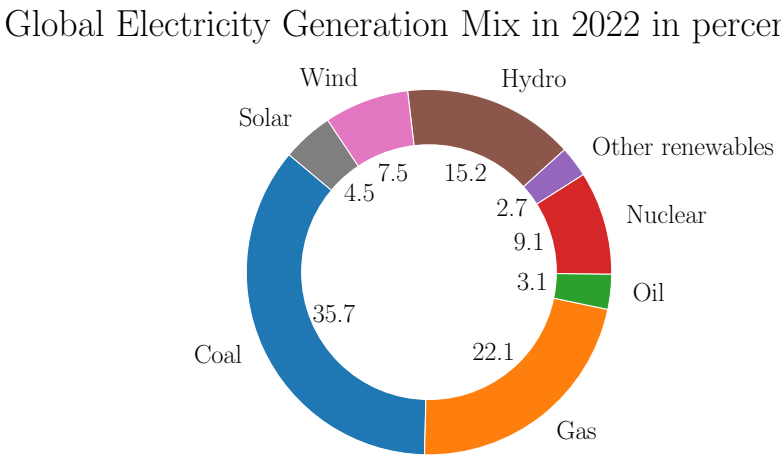


Figure 1.2: Electricity generation sources mix in 2022 [6]

The obstacles impeding the advancement of wind energy can be systematically categorized into several distinct domains, encompassing social, economic, and technical challenges [7, 8, 9]. Each of these areas presents unique limitations that must be addressed to facilitate the broader integration and acceptance of wind energy as a key component of the global energy mix. Veers et al. in [8, 10, 11] mapped out the grand challenges and recommendations that wind energy research needs to tackle and follow to ease the wind energy transition and its substitution for fossil fuel emitting energy sources. One of the recommendations is:

> "Novel approaches to the design process, investigating *probabilistic methods*, *machine learning*, the use of reduced-order *surrogate models*, and other techniques to address the *stochastic* nature of the environment and incorporate inherent uncertainty in the manufactured structures need to be investigated and brought into common practice. A new design process that is better able to capture the physics of inflows, scales, uncertainty, and high dimensionality and to leverage advanced simulation capabilities is critical to building turbines and plants suited to the requirements and limitations of each site." Veers et al.[11]

The research presented in this dissertation follows the aforementioned recommendation. In this research, I developed surrogate models not only for design but also for utilizing them in wind turbine operation and maintenance.

1.1 Background and motivation

Veers et al. in [11] introduces the future of wind energy technology goals. In short, the goals are defined as:

- Optimizing the economic aspect of producing electricity through wind power.
- Minimizing the adverse social and environmental effects of wind energy technology.
- Integrating wind energy into a renewable-dominated energy mix.
- Shortening the design cycle empowered by simulation technology through "numerical prototyping".

These four goals are interrelated. Optimizing the economic aspect often leads to an increase in turbine size, which impacts the social, environmental, and wind energy's role in the energy mix. Additionally, larger rotors face more stochastic and complex inflow, particularly in the floating offshore environment. Consequently, simulations need to account for a broader range of inherent stochasticity in both wind and waves, necessitating a large set of mid to high-fidelity simulations. A viable solution for our simulations and numerical prototyping is the development of Surrogate

Models (SM).

Some SMs originate from the field of Uncertainty Quantification [12]. In Uncertainty Quantification (UQ), we run Monte Carlo Simulation to understand the distribution of the model output. This involves taking samples from the input distribution, running the model on these samples to obtain the output, and then analyzing the resulting output distribution. However, this is a time and computationally-intensive process. An alternative approach is to construct statistical models, or SMs, based on a smart sampling method of the inputs and then fitting a regression model to the outputs. SMs are essentially regression models that can take various forms, such as polynomials, or a ML algorithm [13].

SMs have gained popularity in wind energy research in recent years. In the wind energy domain, surrogate models have been developed for turbine design and optimization, as well as for wind farm design, operation, and maintenance. Table 1.1 provides an overview of a selection of recent research in this field. The provided list, while detailed, is not exhaustive highlighting the dynamic and evolving nature of the subject matter and the potential for ongoing inquiry and discovery.

Table 1.1: SM in wind energy research overview. Abbreviations: Polynomial Chaos Expansion (PCE), Machine Learning (ML), Artificial Neural Network (ANN), Gaussian Process Regression (GPR), Supervisory Control and Data Acquisition (SCADA), Conditional Variational Autoencoders (CVAE), Long Short-Term Memory (LSTM)

Reference	**Methodology**	**Application**
Dimitrov et al. [14]	PCE, quadratic response surface, Kriging, importance sampling, nearest-neighbour interpolation	Fatigue load estimation
Murcia et al. [15]	PCE	Wind farm layout optimization
Dimitrov [16]	ML - ANN	Wake induced loads estimation
Ananthan et al. [17]	ML - ANN	Wind turbine load estimation

Reference	Methodology	Application
Schröder et al. [18]	ML - ANN	Loads and reliability in a wind farm
Avendaño-Valencia et al. [19]	GPR	Fatigue estimation of a turbine in wake
Mylonas et al. [20]	ML - CVAE	Probabilistic blade fatigue estimation based on SCADA
Dimitrov and Natarajan [21]	ML-ANN	Wind farm optimization
Jasa et al. [22]	ML - ANN	Blade design by controlling airfoil shape
Harrison-Atlas et al. [23]	ML - ANN	Wind farm control strategy
Dimitrov and Göçmen [24]	ML - LSTM	Load time series prediction

The literature highlighted in Table 1.1 aims to efficiently map inputs (e.g. wind farm layout, inflow wind characteristics, rotor speed) to outputs (e.g. extreme loads, fatigue loads, power), eliminating the need for costly numerical simulations of varying fidelity levels. The common denominator of this mapping is the input consists of two or more variables. This means the wind time series as the sole input to the SM has not been utilized in previous research. The challenge here is considering the methodologies to build SM on sequential data, wind time series do not provide enough information, which is why a large number of simulations is required [16, 18]. Thus, direct mapping of wind time series to extreme and fatigue loads on wind turbines has not been extensively researched. However, recent developments in sequential ML methods [25, 26] provide an opportunity to explore the possibility of mapping wind time series to wind turbine loads.

1.2 Objective

In this dissertation, I aim to answer the following questions:

- What kind of the SM is required to map wind times series to the extreme and fatigue load on a wind turbine?
- What is the optimum sampling method to build the SMs?
- How can we extend the applicability of the developed SM with minimum effort?
- How can we use the publicly available wind turbine models to help turbine health assessment when the manufacturer model is unavailable?

The capability to correlate wind time series with load data presents considerable benefits, especially in the application of proposed SMs within the realms of digital twins, control systems, and practical scenarios. Concurrently, in research endeavors where Supervisory Control and Data Acquisition (SCADA) data was accessible, models of wind turbines were readily available, as referenced in studies [27, 18, 19, 28, 24]. The use of publicly available models for evaluating the health of wind turbines offers a strategic advantage by bypassing the intricacies related to intellectual property rights, thereby providing significant value to both researchers and turbine owners.

This doctoral research initially extended the work of Fluck [29], focusing on stochastic design and optimization utilizing SMs. Fluck developed an intrusive PCE method to correlate wind conditions with load data. The intrusive approach, however, faced two major challenges: a) the necessity to modify the existing model into a stochastic framework, and b) the issue known as the "curse of dimensionality." To align with prevailing industrial practices and tools, the research pivoted towards exploring non-intrusive PCE techniques. Despite successfully developing an SM based on this methodology, the curse of dimensionality remained a significant obstacle, as was the case with the intrusive approach.

In my endeavour to map wind time series to load data using PCE, a critical issue arose: the outputs lacked auto-correlation inherent to wind and load time series. Subsequently, the research explored the Proper Orthogonal Decomposition-Polynomial Chaos Expansion methodology [30]. This method entailed decomposing the result space into stochastic and temporal domains, with PCE managing the stochastic aspect and Proper Orthogonal Decomposition (POD) addressing the temporal domain to address the auto-correlation in the output time series. Nevertheless, this approach fell short of expectations, hindered by orthogonality constraints and the excessive

number of POD modes required, leading to a dimensionality problem akin to previous methodologies.

Following the challenges encountered with the POD-PCE approach, the research underwent two pivotal evolutions:

- A redirection of research focus from the stochastic design space for machine optimization to applications in operation and maintenance. This shift was precipitated by the recurring dimensionality issues inherent in various methodologies and the necessity for extensive simulation sets, which required frequent updates during each optimization cycle. Additionally, the availability of operational data during the research was a decisive factor.
- A methodological transition from reliance on polynomial-based SMs to the adoption of ML based SMs which are generally non-linear and non-parametric. The initial use of PCE for constructing SMs was conceived a decade ago, a period when ML methodologies and tools were not as advanced as they are today. The surge in ML tools and their increased accessibility and user-friendliness in recent years motivated and facilitated this change.

Consequently, the research began employing sequential ML methods to map wind time series to load time series. However, given that the wind turbine industry primarily focuses on analyzing fatigue or extreme loads resulting from load time series, the research objectives were realigned towards mapping wind time series directly to fatigue or extreme loads. This emerging trend is corroborated by recent studies in the field, as noted in the work of Dimitrov et al.[14, 16, 18, 24].

The overarching goal is to develop a versatile and efficient SMs capable of translating wind time series into fatigue and extreme load data on wind turbines, and wind turbines generated power. The potential applications for this SM are diverse, encompassing:

- Wind farm design and optimization, where the SM can adeptly account for wake effects from neighbouring turbines.

- Wind turbine health assessment, enabling the SM to provide insights into the structural integrity of turbines.
- Predictive control of wind turbines, utilizing the SM to anticipate incoming inflows and subsequently reduce turbine loads.
- Enhancement of wind turbine Digital Twin, where an expedient SM can refine the tuning procedures of Digital Twin (DT)s.

It is crucial to remember that this list is not exhaustive and can include more options.

1.3 Research contribution

This dissertation aims to develop and build SMs for mapping wind time series to the loads on a wind turbine, with applications in wind farm design and optimization, and wind turbine health assessment. The research contributes the following advancements to the state-of-the-art knowledge:

1. Development of a non-intrusive PCE-based SM for time marching output of Blade Element Momentum simulations.
   (a) Demonstration of rotor loads sectional statistics convergence in the output for a reference case with $48K$ simulations.
   (b) Evidence that a limited number of sectional SMs are sufficient for accurate load estimation.
   (c) Exploration of extreme load sensitivity to the number of MCS in the SM.
2. Development of a sequential ML-based SM to map wind time series to Damage Equivalent Load.
   (a) Construction of an extensive simulation database using Sobol's sampling.
   (b) Validation of the developed SM's ability to accurately predict DEL.
   (c) Illustration of the SM's capability in understanding wind turbine structural dynamics.

(d) Development of a simplified wake model and demonstration that the SM can accurately predict DEL for turbines in wake conditions using TL.

3. Development of a Gaussian Process Regression-based SM, trained on a database assimilating SCADA data, to map wind speed to DEL and load averages.

   (a) Creation of an extensive database assimilating SCADA measurements based on a publicly available wind turbine model.

   (b) Confirmation of accuracy of the trained GPR in predicting test DEL.

   (c) Accuracy of the GPR's validated against the SCADA loads and tower top acceleration.

   (d) Demonstration that GPR, using a publicly available model and minimal SCADA measurements, can serve as a primary tool for wind turbine health assessment.

1.4 Dissertation outline

This dissertation presents a series of three interconnected manuscripts, each published or submitted to academic journals. Collectively, they form a cohesive narrative, offering an in-depth exploration of the research topic. Each chapter includes an abstract, introduction, methodology, and conclusion, providing a comprehensive understanding of the research. The nomenclature is consistently explained in the text across all chapters, with the same abbreviations employed throughout the dissertation. In Chapter 2, a few tables have been reformatted from the original manuscript to fit the typesetting of this dissertation. The structure of the dissertation is as follows.

Chapter 2 builds upon the work of Fluck, advancing a non-intrusive PCE for BEM, with a focus on unsteady optimization. This chapter presents an analysis of unsteady wind aerodynamics, revealing stationary characteristics using standard turbulence models. The analysis is based on outputs from NREL 5MW reference machine BEM simulations. I introduce a non-intrusive PCE method to construct a SM for the rotor thrust and torque statistics at each time step, enhancing the accuracy and efficiency of extreme statistics estimation.

Chapter 3, the core chapter of the dissertation, represents the culmination of two significant research evolutions. It proposes a data-driven approach for calculating DEL on wind turbines, utilizing a SM based on aeroelastic simulation outputs. The chapter explores the use of sequential ML techniques to map wind speed time series to DEL, enabling rapid and efficient DEL calculations. The versatility of the trained surrogate models is demonstrated, including their ability to digest wind turbine structural dynamics and application to turbines in wake scenarios and the use of transfer learning to refine prediction accuracy.

Chapter 4 details the construction of a GPR model, trained on a combination of SCADA measurements and synthetic data. Addressing the challenge of wind turbine structural health assessment, this chapter begins by creating a hybrid database integrating aero-servo-elastic simulations from publicly available wind turbine models and tools with SCADA data. GPR models are then developed to predict wind turbine loads, with validation against both hybrid simulations and SCADA measurements. The results underscore the potential of GPR models, trained on publicly available models and limited SCADA data, as valuable tools for wind turbine health assessment.

The dissertation concludes with Chapter 5, summarizing the main findings, discussing the limitations and assumptions of the research, and proposing directions for future work in this area.

Bibliography

[1] Katherine Calvin, Dipak Dasgupta, Gerhard Krinner, Aditi Mukherji, et al. IPCC, 2023: Climate Change 2023: Synthesis Report. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change [Core Writing Team, H. Lee and J. Romero (eds.)]. IPCC, Geneva, Switzerland.

[2] Home :: Media and Climate Change Observatory. URL `https://sciencepolicy.colorado.edu/icecaps/research/media_coverage/index.html`.

[3] Anne M. van Valkengoed and Linda Steg. Climate anxiety is about more than just personal risks. 13(7):591–591. ISSN 1758-6798. doi: 10.1038/s41558-023-01718-0.

[4] Alec Tyson. How important is climate change to voters in the 2020 election? URL `https://www.pewresearch.org/short-reads/2020/10/06/how-important-is-climate-change-to-voters-in-the-2020-election/`.

[5] Hannah Ritchie, Pablo Rosado, and Max Roser. Emissions by sector. *Our World in Data*, 2020. https://ourworldindata.org/emissions-by-sector.

[6] Hannah Ritchie and Pablo Rosado. Electricity mix. *Our World in Data*, 2020. https://ourworldindata.org/electricity-mix.

[7] GAM Van Kuik, Joachim Peinke, Rogier Nijssen, Denja Lekou, Jakob Mann, Jens Nørkær Sørensen, Carlos Ferreira, Jan-Willem van Wingerden, David Schlipf, Pieter Gebraad, et al. Long-term research challenges in wind energy – a research agenda by the European Academy of Wind Energy. 1(1):1–39. ISSN 2366-7443. doi: 10.5194/wes-1-1-2016.

[8] Paul Veers, Katherine Dykes, Eric Lantz, Stephan Barth, Carlo L. Bottasso, et al. Grand challenges in the science of wind energy. 366(6464):eaau2027, . doi: 10.1126/science.aau2027.

[9] Julia Kirch Kirkegaard, David Philipp Rudolph, Sophie Nyborg, Helena Solman, Elizabeth Gill, Tom Cronin, and Mary Hallisey. Tackling grand challenges in wind energy through a socio-technical perspective. 8(7):655–664. ISSN 2058-7546. doi: 10.1038/s41560-023-01266-z.

[10] Paul Veers, Katherine Dykes, Sukanta Basu, Alessandro Bianchini, Andrew Clifton, Peter Green, Hannele Holttinen, et al. Grand Challenges: Wind energy research needs for a global energy transition. 7(6):2491–2496, . ISSN 2366-7443. doi: 10.5194/wes-7-2491-2022.

[11] Paul Veers, Carlo L. Bottasso, Lance Manuel, Jonathan Naughton, Lucy Pao, Joshua Paquette, Amy Robertson, et al. Grand challenges in the design, manufacture, and operation of future wind turbine systems. 8(7):1071–1131, . ISSN 2366-7443. doi: 10.5194/wes-8-1071-2023.

[12] Bruno Sudret. *Uncertainty propagation and sensitivity analysis in mechanical models–Contributions to structural reliability and stochastic spectral methods.* PhD thesis, 2007.

[13] Reza Alizadeh, Janet K. Allen, and Farrokh Mistree. Managing computational complexity using surrogate models: A critical review. *Research in Engineering Design*, 31(3):275–298, 2020. ISSN 1435-6066. doi: 10.1007/s00163-020-00336-7.

[14] Nikolay Dimitrov, Mark C. Kelly, Andrea Vignaroli, and Jacob Berg. From wind to loads: Wind turbine site-specific load estimation with surrogate models trained on high-fidelity load databases. *Wind Energy Science*, 3(2):767–790, 2018. ISSN 2366-7443. doi: 10.5194/wes-3-767-2018.

[15] Juan Pablo Murcia, Pierre-Elouan Réthoré, Nikolay Dimitrov, Anand Natarajan, John Dalsgaard Sørensen, Peter Graf, and Taeseong Kim. Uncertainty propagation through an aeroelastic wind turbine model using polynomial surrogates. *Renewable Energy*, 119:910–922, 2018. ISSN 0960-1481. doi: 10.1016/j.renene.2017.07.070.

[16] Nikolay Dimitrov. Surrogate models for parameterized representation of wake-induced loads in wind farms. *Wind Energy*, 22(10):1371–1389, 2019. ISSN 1099-1824. doi: 10.1002/we.2362.

[17] Shreyas Ananthan, Ganesh Vijayakumar, and Shashank Yellapantula. A DNN surrogate unsteady aerodynamic model for wind turbine loads calculations. *Journal of Physics: Conference Series*, 1618(5):052060, 2020. ISSN 1742-6596. doi: 10.1088/1742-6596/1618/5/052060.

[18] Laura Schröder, Nikolay Krasimirov Dimitrov, and David Robert Verelst. A surrogate model approach for associating wind farm load variations with turbine failures. *Wind Energy Science*, 5(3):1007–1022, 2020. ISSN 2366-7443. doi: 10.5194/wes-5-1007-2020.

[19] Luis David Avendaño-Valencia, Imad Abdallah, and Eleni Chatzi. Virtual fatigue diagnostics of wake-affected wind turbine via Gaussian Process Regression. *Renewable Energy*, 170:539–561, 2021. ISSN 0960-1481. doi: 10.1016/j.renene.2021.02.003.

[20] Charilaos Mylonas, Imad Abdallah, and Eleni Chatzi. Conditional variational autoencoders for probabilistic wind turbine blade fatigue estimation using Supervisory, Control, and Data Acquisition data. *Wind Energy*, 24(10):1122–1139, 2021. ISSN 1099-1824. doi: 10.1002/we.2621.

[21] Nikolay Dimitrov and Anand Natarajan. Wind farm set point optimization with surrogate models for load and power output targets. *Journal of Physics: Conference Series*, 2018(1):012013, 2021. ISSN 1742-6596. doi: 10.1088/1742-6596/2018/1/012013.

[22] John Jasa, Andrew Glaws, Pietro Bortolotti, Ganesh Vijayakumar, and Garrett Barter. Wind turbine blade design with airfoil shape control using invertible neural networks. *Journal of Physics: Conference Series*, 2265(4):042052, 2022. ISSN 1742-6596. doi: 10.1088/1742-6596/2265/4/042052.

[23] Dylan Harrison-Atlas, Ryan N. King, and Andrew Glaws. Machine learning enables national assessment of wind plant controls with implications for land use. *Wind Energy*, 25(4):618–638, 2022. ISSN 1099-1824. doi: 10.1002/we.2689.

[24] Nikolay Dimitrov and Tuhfe Göçmen. Virtual sensors for wind turbines with machine learning-based time series models. *Wind Energy*, 25(9):1626–1645, 2022. ISSN 1099-1824. doi: 10.1002/we.2762.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html`.

[26] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling, 2018.

[27] Nikolay Dimitrov and Anand Natarajan. From SCADA to lifetime assessment and performance optimization: How to use models and machine learning to extract useful insights from limited data. *Journal of Physics: Conference Series*, 1222(1):012032, 2019. ISSN 1742-6596. doi: 10.1088/1742-6596/1222/1/012032.

[28] Laura Schröder, Nikolay Krasimirov Dimitrov, David Robert Verelst, and John Aasted Sørensen. Using Transfer Learning to Build Physics-Informed Machine Learning Models for Improved Wind Farm Monitoring. *Energies*, 15(2): 558, 2022. ISSN 1996-1073. doi: 10.3390/en15020558.

[29] Manuel Fluck. *Stochastic Methods for Unsteady Aerodynamic Analysis of Wings and Wind Turbine Blades*. PhD thesis, 04 2017.

[30] E. Jacquelin, N. Baldanzini, B. Bhattacharyya, D. Brizard, and M. Pierini. Random dynamical system in time domain: A POD-PC model. *Mechanical Systems and Signal Processing*, 133:106251, 2019. ISSN 0888-3270. doi: 10.1016/j.ymssp.2019.106251.

Chapter 2

Surrogate models for the blade element momentum aerodynamic model using non-intrusive Polynomial Chaos Expansions

This chapter was first published in the Wind Energy Science journal as:

Haghi, R. and Crawford, C.: Surrogate models for the blade element momentum aerodynamic model using non-intrusive polynomial chaos expansions, Wind Energ. Sci., 7, 1289–1304, https://doi.org/10.5194/wes-7-1289-2022, 2022.

Author contribution

RH developed the necessary computer code and wrote the paper in consultation with and under the supervision of CC.

Abstract

In typical industrial practice based on IEC standards, wind turbine simulations are computed in the time domain for each mean wind speed bin using a few number of unsteady wind seeds. Software such as FAST, BLADED or HAWC2 can be used to capture the unsteadiness and uncertainties of the wind in the simulations. The statistics of these aeroelastic simulations output are extracted and used to calculate fatigue

and extreme loads on the wind turbine components. The minimum requirement of having six seeds does not guarantee an accurate estimation of the overall statistics. One solution might be running more seeds; however, this will increase the computation cost. Moreover, to move beyond Blade Element Momentum (BEM)-based tools toward vortex/potential flow formulations, a reduction in the computational cost associated with the unsteady flow and uncertainty handling is required. This study illustrates the unsteady wind aerodynamic statistics' stationary character based on the standard turbulence models. This character is shown based on the output of National Renewable Energy Lab (NREL) 5MW reference machine Blade Element Momentum (BEM) simulations. Afterwards, we propose a non-intrusive Polynomial Chaos Expansion (PCE) to build a surrogate model of the loads' statistics, the rotor thrust and torque, at each time step, to estimate the extreme statistics more accurately and efficiently.

2.1 Introduction

The process of calculating loads on wind turbine components is one of the core parts of wind turbine aerodynamic and structural design and optimization. In the last few decades, international organizations have developed different aeroelastic codes such as Fatigue, Aerodynamics, Structures, and Turbulence (FAST) [1], BLADED [2] and HAWC2 [3] to accurately calculate load time series based on the standardized or site-specific environmental conditions. Engineers and researchers use wind turbine aeroelastic simulation output statistics to calculate extreme and fatigue loads on wind turbine structures and estimate the unsteady power. To take into account the randomness in the unsteady wind, according to IEC standards [4], the simulation process must use a semi Monte Carlo (MC) method. Therefore, a full simulation set should include a limited number of seeds for generating multiple wind speed time series of $600s$.

In normal practice, for each mean wind speed, at least six different seeded unsteady wind time series are required as the minimum to take into account the uncertainties. This limited number of unsteady simulations does not yield an entirely accurate estimation of the statistics. Gradient-based optimization algorithms may not deal with these inaccuracies well. One option to solve this problem is running more seeds, which will increase the computational cost. The increase in computational cost will play a more critical role in our decision making if we want to move towards vortex [5] and

potential flow codes for load calculations, as they require more computation resources inherently. An alternative approach to direct simulation is to use a surrogate model that can provide us with an accurate statistical estimation set based on a limited number of simulations.

The origin of the *surrogate model* lies in Uncertainty Quantification (UQ) analysis [6]. There are many uncertainty quantification implementations in wind energy. More specifically, surrogate models show much potential in wind farm load estimation, wind turbine optimization or reliability analysis. Many researchers have investigated these potentials. For example, [7, 8, 9, 10] used surrogate models to estimate the loads on a wind turbine based on the stochastic variables gross parameters such as turbulence intensity, mean wind or wind direction. [11, 12, 13] used surrogate models for uncertainty propagation through the wind turbine models. More recently, the surrogate models have been used for the wind turbines reliability assessments [14, 15]. Also, [16] and [17] showed the application of surrogate model in wind turbine optimization. However, very few have looked at building a surrogate model of the aerodynamic model of wind turbine based on the random phases as the input. Fluck and Crawford showed an initial attempt to build a surrogate model based on intrusive Polynomial Chaos Expansion (PCE) on simple lifting line and BEM models [19, 18]. As they were quickly faced with *curse of dimensionality*, they showed it is possible to reduce the number of random variables in Veers' unsteady wind model significantly. Afterwards, they used this reduced dimension wind model to propagate stochasticity through a simple lifting line [21] or BEM [18] model. However, with intrusive PCE it is necessary to change the model implementation fundamentally to incorporate the random variables [6]. This might work for a simple model, but when we want to utilize commercially available aeroelastic codes, this will be challenging or even impossible.

This paper's goal is to build a non-intrusive PCE surrogate model of a deterministic aerodynamic model driven by stochastic unsteady wind. This study's implemented aerodynamic model takes wind time series as input and calculates thrust and torque on a NREL 5MW turbine rotor using BEM. The motivation is to build a surrogate model based on a limited amount of simulation data to estimate the statistics of the aerodynamic model output at each time step of the time series quickly. Having this surrogate model at hand helps us explore and experience the opportunities it can provide. This output guides future research in the surrogate model realms for us in

the long run. The surrogate model investigation presented is an exploration of the potential benefits and limitations of PCE-based time-domain surrogate models, to help researchers and practitioners develop future surrogate modeling approaches.

As the surrogate models are inherently cheap to run, we take this surrogate model through a MCS large number of times. The input of these MCS are the samples drawn from the uniform random variables. The unsteady wind generator uses the same random variable distribution to make sure the generated time series will match a Gaussian process [20]. This process is presented in Figure 2.1 schematically. By this method, we can reduce the computational cost and time for the aerodynamic simulation, without compromising the validity of the results. One can interpret this model as a tool to map the input distribution (in this case, an uniform distribution of random seeds-phases) to the output distribution (in this case, distribution of thrust and torque on the rotor).

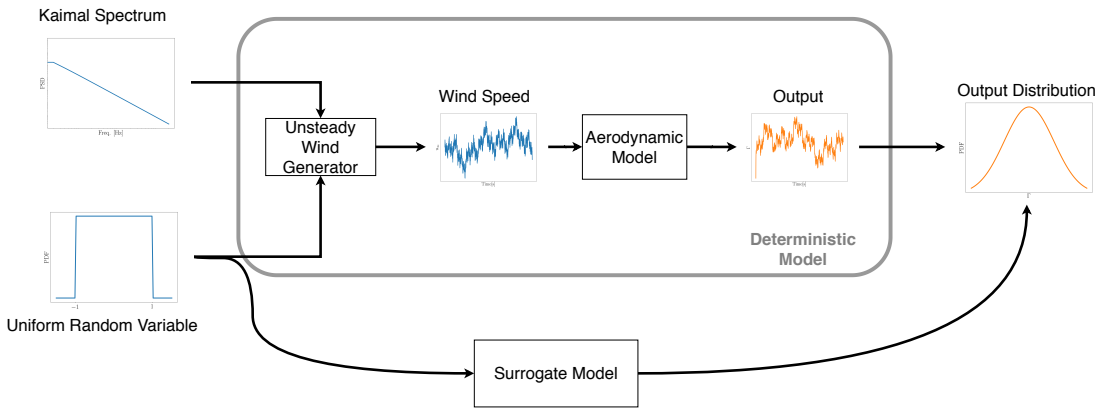


Figure 2.1: The common deterministic process of aerodynamic modeling vs the suggested surrogate model method schematic flow chart.

Fitting a surrogate model at each time step of $600s$ of the aerodynamic output times series, using the random phases (Figure 2.1), is computationally expensive and redundant to current practice. Therefore, we start by showing that the aerodynamic simulation results based on Veer's reduced model [22] statistically converges. We also show that the unsteady wind aerodynamic process's statistical properties are constant in time (stationary process). Therefore, by keeping the computational effort the same, it is possible to run more simulations while shortening the length of the

simulations. Furthermore, more simulations with the same computational effort give us the chance to fit higher degree PCEs, which provides a more accurate estimation of the statistics. We build four different PCEs surrogate models, with four different polynomial degrees (degrees two to five) to pick the best in terms of accuracy and computational cost trade-off. These surrogate models have been used for MCS for a large number of runs (cheaply). The results of the MC runs of the surrogate models are compared with 48000 unsteady wind aerodynamic simulation results. In this case, the simulation results are the thrust and torque forces induced on the NREL 5MW in an unsteady wind. We compare the results by looking at the thrust and torque distribution from both the deterministic and surrogate models. Finally, we show how the extreme loads extracted from the MCS can converge to the extreme loads from the results from 48000 unsteady wind aerodynamic simulation results.

This paper is organized as follows. Section 2.2 describes the unsteady wind generation and aerodynamic model. Sections 2.3 and 2.4 explain the and statistical elements and PCE method used in the study. Next, in section 2.5, we set out the approach to tackle the challenge. Section 2.6 provides the BEM simulation results, convergence of the sectional statistics the PCE fit on the sectional statistics and the emulations output. At the end of the results section, we discussed the accuracy and efficiency of the surrogate models developed in this study. This paper concludes in Section 2.7, reiterating the key findings of the study and offering suggestions for fruitful future work in the area of wind turbine surrogates.

2.2 Models

This section provides an overview of the unsteady wind generation basics and aerodynamic model used in this study.

2.2.1 Reduced Veers unsteady wind model

One famous unsteady wind model in the wind turbine community is the Veers model [20]. The history of the model goes back to the late 80s' and has a long record in wind turbine load calculation practice. Very briefly, Veers unsteady wind model is inherently an inverse Fourier transformation. The 1-D unsteady wind time series at

location P is generated via:

$$u_{\infty}(t_n) = \sum_m \sqrt{S_m} e^{i(\omega_m t_n + 2\pi\xi_m)} \tag{2.1}$$

For this inverse Fourier transformation in Eq (2.1), the frequencies ω_m and are taken from the Kaimal spectrum (Figure 2.1). The random phases are based on the independent random variable ξ_m drawn from a uniform distribution over $[0, 1]$. Finally, the amplitude S_m is specified based on the power spectrum at the frequencies ω_m [23].

In load calculation practice, Veers' model for the unsteady wind is the method to generate turbulent boxes, commonly implemented in TurbSim [24]. The method is briefly explained in [22] and extensively in [20]. To make the unsteady wind in TurbSim, this method uses a large number of random variables on the order of thousands is required [24]. This large number of random variables pushes the surrogate model problem into the *curse of dimensionality* very quickly. Therefore, building a PCE surrogate model is almost impossible. To tackle this issue, Fluck and Crawford [21] showed that using only ten uniformly distributed independent random variables with ten frequencies logarithmically sampled from the Kaimal spectrum [20] is enough for building unsteady wind time series. This *Reduced Veers' model* generated unsteady wind that can capture the same level of randomness and probability distribution as the full model. This study used this reduced Veers model to generate unsteady wind time series. This method does not lead to a model that fully replaces high fidelity Turbsim outputs but rather a surrogate model necessity to study trade-offs of various accuracy and assumption aspects.

The randomness in the generated unsteady wind comes from the ten random variables, ϕ_j, in the $\boldsymbol{\xi}$ vector describing the frequency components' phases $2\pi\xi_m$ in the reduced Veers model in Eq. (2.1) [22]. Based on the Veers method [20] and in TurbSim [24], the employed sampling method is a pseudo-Random Number Generator (pRNG) which is the basis of MCS. However, the problem with this way of sampling for MCS is the lack of control over the random variables' domain as it may fill some voids in the domain and may leave some of it empty [25]. Therefore, the random domain may not be filled evenly for the same reason. For this study, a low discrepancy Quasi Monte Carlo (QMC) sampling method, namely the Sobol sequence

[26] is used to draw samples from the random variables in this work to calculate the PCE coefficients via the point collocation method. It is worth mentioning that we did not consider any uncertainties in the PCE coefficients. The main reasons to select the Sobol sequence over other sampling methods are the samples' consistency and computational efficiency [27]. A custom random wind generator based on the reduced Veers model used these samples to generate unsteady wind fields.

2.2.2 Aerodynamic model

The aerodynamic model for this study is a BEM model [23] with frozen wake based on the work of [28]. This non-linear BEM model is used to run simulations on a NREL 5MW [29] rotor to calculate thrust and torque on the rotor. For the simulations, the rotor speed was kept constant based on the mean wind speed of the simulations. Also, the pitch angle was set according to the data provided in [29]. There is no controller involved in the simulations. The unsteady wind defined in the previous section is set at 100m hub height and remains the same on the rotor. The Python package for BEM is `bemused` [28]. The NREL 5MW model characteristics and properties are extracted from [29]. The model employed in this study for the simulations and surrogate model is essentially equivalent to the NREL 5MW model in any wind turbine aerodynamic code (e.g. FAST, BLADED and HAWC), but nicely formulated in Python. The model and analysis code used in this work has been previously verified against the NREL 5MW full model using BLADED by Lupton [30].

2.3 Statistical convergence metric

For this study, we want to investigate the null hypothesis that cross-sectional statistics (statistics at each time step) of a combination of a large number of aerodynamic simulation outputs are similar. In other words, we want to investigate if the statistical properties of the output at each time step converge as a function of the number of simulations (stationary process) for the non-linear stochastically autocorrelated system. Figure 2.2 presents a generic example of distributions (histogram fits) at each time step for a set of realizations of one random process. (The figure shows a schematic plot; therefore, histograms and fitted distributions *do not* represent the

plotted trajectories.)

To show that the sectional statistics of a large number of simulations are convergent, we need a metric to quantify the difference between the distributions at each time step. There are different metrics for this purpose [31]; for this study, we use *Hellinger Distance (HD)* [32] as a metric due to its ease of application and interpretation. The HD is the metric to quantify how similar two probability distributions are to each other. The distance is zero if they are the same and is one when the two distributions are disjoint. The HD for two *discrete* probability distributions P and Q, which have an equal number of bins, can be formulated as:

$$H(P,Q) = \frac{1}{\sqrt{2}}\sqrt{\sum_{i=1}^{k}(\sqrt{p_i} - \sqrt{q_i})^2} \tag{2.2}$$

In Eq.(2.2), p_i and q_i are the probabilities for P and Q at every bin. In our case, to make the comparison fair, not only the number of bins are the same, but also the bin width is the same for both P and Q. This assures us that there is no artificial distance reduction in the results.

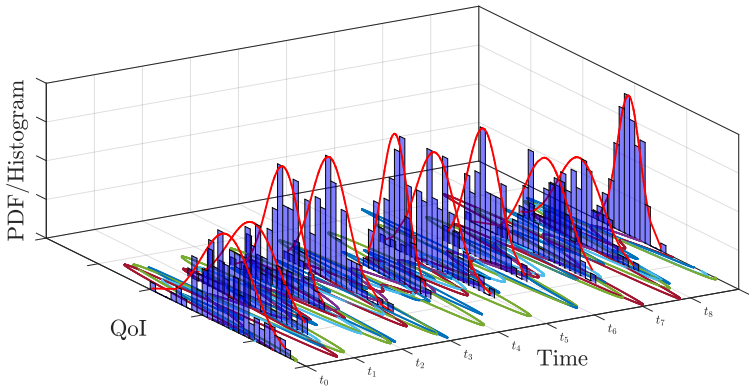


Figure 2.2: A schematic drawing presenting possible distributions at each time step based on a set of time trajectories for a Quantity of Interest (QoI)

In this study, we use HD to show the cross-sectional statistics changes for a large number of simulations is minimal. Therefore, we can shorten the simulations without losing the accuracy of the sectional statistics. We also use the HD as an error metrics

to compare the accuracy of the MCS with the reference case. The reason behind choosing this measure, instead of simply looking at the mean and stand deviation difference, is the distribution of the aerodynamic model output. As the next section we will show, the distribution of the aerodynamic output is a Weibull distribution. Therefore, comparing mean and standard deviation would not provide us the full statistical picture.

2.4 Polynomial Chaos Expansion Fundamentals

Uncertainty propagation of mathematical models has been the subject of many studies in the last thirty years. One method to propagate uncertainty is using models of models, called surrogate models. A surrogate model is a cheap-to-run approximation of the actual model [33]. Among surrogate models, the Polynomial Chaos Expansion (PCE) has gained attention especially after the work of [34] and [35]. PCE is a method that uses a variable described by its statistical distribution (random variable ξ) and projects the model onto a basis of polynomials. This way, the uncertainty can be propagated through the model with a limited number of simulations [36]. In other words, PCE is a technique to estimate the response of a mathematical or numerical model based on a series of orthogonal polynomials, which are functions of a random variable $\boldsymbol{\xi}$. The solution is expanded and described in stochastic space spanned by $\boldsymbol{\xi}$ and the associated polynomial basis.

The main reasons to use PCE instead of other surrogate model methods are; a) with minimum computational effort, one can extract statistical moments directly from PCEs; b) PCEs are easy to integrate into deterministic linear and non-linear mathematical models; c) one can build PCE surrogate model by treating the model as a black box [37, 38] using a non-intrusive formulation.

In order to illustrate the application of PCE surrogates, assume $Y(t_n) = \mathrm{M}(t_n, \boldsymbol{\xi})$ where t_n is time step n, and $\boldsymbol{\xi}$ is the random variable vector, $\mathrm{M}(t_n, \boldsymbol{\xi})$ is our deterministic time marching mathematical model and $Y(t_n)$ is the output of the model at time step n. Therefore, the stochastic output of the model $Y(t_n, \boldsymbol{\xi})$ can be expanded as:

$$Y(t_n, \boldsymbol{\xi}) = \sum_{i=1}^{\infty} y_i(t_n)\Psi_i(\boldsymbol{\xi}) \tag{2.3}$$

where $y_i(t_n)$ are PCE coefficients at *each time step* and $\Psi_i(\boldsymbol{\xi})$ is a member of an orthogonal polynomial class. These polynomials are orthogonal with respect to the probability space of random variable $\boldsymbol{\xi}$. The selection of the polynomial type is a function of the probability distribution on the random variable $\boldsymbol{\xi}$. For example, if a random variable $\boldsymbol{\xi}$ has a normal distribution, then a Hermite polynomial is selected [39].

The polynomials do not necessarily need to be selected from the specific polynomials family as long as they are orthogonal polynomials. For instance, [18] showed exponential components worked best for their purposes. As the randomness in this study comes in the form of a uniform distribution for wind frequency component phases ϕ_j, the surrogate model is based on the Legendre polynomials [35]. In practice, the PCE summation in Eq (2.3) is truncated at a reasonably high order p. The task of fitting the expansion in Eq (2.3) is finding the coefficients $y_i(t)$. There are two main approaches to solve this problem:

- the *intrusive* approach where the model is projected on the orthogonal polynomials using a Galerkin projection [40]. This approach requires building a detailed stochastic model from the deterministic model governing equations. The intrusive approach was used by [21, 18] to build a surrogate model on lifting line and BEM models.
- the *non-intrusive* approach allows calculating the PCE coefficients from a series of deterministic model evaluations. This approach considers the model as a black box and does not require any model modification [6, 41]. There are two sub-categories to calculate the coefficients, namely *simulation methods* and *quadrature methods* [6].

The presented work uses the *non-intrusive* approach to calculate the PCE coefficients. In the non-intrusive approach category we primarily used a *simulation method* to calculate the PCE coefficients. In mathematical form, the output of the aerodynamic model $\mathrm{M}_{aero}(t_n, \boldsymbol{\xi})$ at time step n is thrust $Trt(t_n)$ and torque $Trq(t_n)$. Therefore, one can re-write Eq. (2.3) as:

$$Trt(t_n, \boldsymbol{\xi}) \approx \sum_{i=1}^{m} T_i(t_n)\Psi_i(\boldsymbol{\xi}) \tag{2.4}$$

$$Trq(t_n, \boldsymbol{\xi}) \approx \sum_{i=1}^{m} \tau_i(t_n)\Psi_i(\boldsymbol{\xi}) \tag{2.5}$$

where the goal is to calculate the polynomial coefficients $T_i(t_n)$ and $\tau_i(t_n)$ at each time step. This surrogate model's input is the random variable $\boldsymbol{\xi}$ vector used in the reduced Veers model to generate the unsteady wind. The surrogate model's output is the thrust and torque at a specific time step for which the surrogate model is built. The main difference between Eq. (2.3) and (2.4)(2.5) is the approximation with finite polynomial series expansion, as it is not feasible to take into account an infinite number of polynomials. This work's surrogate model is built employing the Python toolbox `chaospy` [42]. `chaospy` is a numerical tool for uncertainty quantification using different methods, including PCE. For this study, we used the *point collocation* method to calculate the coefficients due to the ease of implementation in the `chaospy` toolbox. This method has been explained well in the literature (see [43], [34], [6]).

2.5 Surrogate Modeling Methodology

In this piece of work, whenever we talk about simulations, we mean aerodynamic simulations in time using the BEM aerodynamic model introduced above. The input of the aerodynamic simulations is what we call wind time series or unsteady wind and is auto-correlated by construct. This wind time series is generated based on the reduced Veers model explained above.

This study starts with running an extensive set of simulations based on the reduced Veers model at $12m/s$ mean wind speed, aerodynamic simulation model and Sobol sampling, explained previously. This wind speed is the rated power wind speed for NREL 5MW. Afterwards, as we have a large number of simulations output in our database, we can show that the thrust and torque statistics with time are not changing significantly. Therefore the statistical process properties at each time step (e.g. mean, standard deviation, etc.) would be significantly similar to other time steps. Knowing the process statistics stays the same in time, we conclude that only building

a single surrogate model, i.e., a single time-step or a few ones, would suffice for our purpose. The accuracy of the PCEs depends on the polynomial degree. However, an increase in the polynomial degree pushes the problem further toward the *curse of dimensionality*. The number of required coefficients to build the surrogate model and the required number of simulations are presented in Table 2.1. Eq (2.6) presents the formula to calculate the number of PCE coefficients. In Eq. (2.6), M is the required number of coefficients, N is the number of random variables, and P is the polynomial order.

$$M + 1 = \frac{(N + P)!}{P!\, N!} \tag{2.6}$$

The table shows we need a large number of simulations to build an accurate PCE. According to Hosder et al. recommendation, twice the number of simulations $M + 1$ can provide acceptable accuracy for the point collocation method [44]. That recommendation is the basis for the number of simulations in this study.

We want to limit the computational cost for a single average wind speed to 6 times $600s$ simulations (in total $3600s$) to be competitive with the standard practice in wind turbine aerodynamic simulation. Combining a large number of simulations and $3600s$ cumulative simulations length leads to a large number of short simulations instead of a few long ones. We kept the simulation's cumulative length at 3600 seconds to make this trade-off fair. This means that as the simulations' length decreases, the number of simulations increases. Sobol sampling is the base of the unsteady wind generation and input to the aerodynamic simulation setup. For every set of the required number of simulations in Table 2.1, the random phases are drawn independently from the rest of the sets. For example, for the second row of Table 2.1, when we need 132 simulations, 132 unique samples of $\boldsymbol{\xi}$ are drawn from the random domain. These $\boldsymbol{\xi}$ have not been used for other simulation sets. By having a large number of data points at each time step, we built a few surrogate models in time and compared the results with the simulations' reference case. For the sake of accuracy, in this study, we do not build any surrogate model based on 1st-degree polynomials.

Another approach to calculate the coefficients is Gaussian Quadrature (GQ). This method has been extensively explained in the literature e.g. [45]. There are also ex-

Table 2.1: The number of coefficients and required data points to calculate the coefficients for 10 random variables and using the point collocation method. This number of coefficients should be calculated for every time step. The last column demonstrates the simulation length for the fitted PCEs as explained in Section 2.6.

Polynomial deg.	No of coeff. + 1	No of req sim	Sims. length to fit PCEs [s]
1	11	22	163.6
2	66	132	27.3
3	268	536	6.7
4	1001	2002	1.8
5	3003	6006	0.6

tensions to GQ referred to as Sparse Gaussian Quadrature (SGQ) methods that seek to reduce the number of simulations required to fit the surrogate (e.g. Smolyak). Our tests show that for a standard GQ method with 10 random variables and polynomial orders 2, 3 and 4 we need 59049, 1048576, 9765625 evaluation points respectively. On the other hand, the Smolyak sparse approach for GQ [45, 46], will reduce the number of evaluation points drastically. We tested the SGQ method for polynomial orders of 2,3 and 4 with 10 uniform random variables and the Smolyak sparse approach for SGQ. The number of the required evaluations for each polynomial order is 221, 1581 and 8761, respectively. We ran the evaluations for the SGQ method, calculated the weights and built the polynomials. However, the results were not as promising as expected. The results from the SGQ method are shown and briefly discussed in Section 2.6.4

For a stationary input (Reduced Veers model), the sample statistics of output converges at the rate of $1/\sqrt{n}$, where n is the number of data points (in this case, 48000 data points at a one-time instance). Consequently, it is possible to *estimate* the statistical parameters of the output distribution by different methods. One possible approach is using the maximum likelihood estimator [47]. A question that then arises is why we go through the complication of building a surrogate model. The research goal we present here is to build a surrogate model of an aerodynamic model, whether the aerodynamic model is simple or complex, with the model capable of resolving the form of the performance statistics, as an alternative to estimation methods. We chose an aerodynamic model that is easy to simulate while complex enough to capture the inherent non-linear behaviour. Hence, the specific aerodynamic example model does

not compromise the validity of the method we introduce here to later more complex aeroelastic simulations with e.g. FAST or BLADED.

2.6 Results

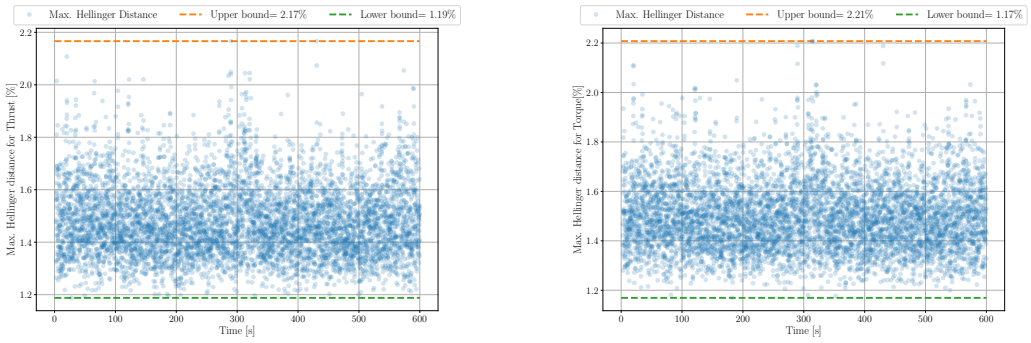
This section presents the results of our numerical experiments. We start by looking at the HD of a large number of aerodynamic simulation output, thrust and torque, and show the distance does not change significantly. Therefore, the sectional statistics are almost the same across time steps. Afterwards, based on that conclusion, we built a number of surrogate models for polynomial orders of 2 to 5 from a limited number of simulations and show the statistics match the reference case. Finally, we show how extracted extreme thrust and torque are comparable with the reference case.

2.6.1 Sectional Statistical Convergence

As mentioned before, in Section 2.5, we started by running a broad set of reference simulations. For this case, we ran 48000 simulations for a $12m/s$ wind speed and turbulence intensity of 0.16. The wind generator code took 48000 samples from a 10 dimensional uniform distribution domain based on the quasi-random sampling method. Each sample is a 10 by 1 vector of $\boldsymbol{\xi}_j$, and we have 48000 of them. 48000 wind speed time histories were generated, and simulations on the aerodynamic models run with a time step of $0.1s$ for $630s$ (in total 6300 time steps per simulation), with $30s$ transient period. We discarded the transient period for all the processes in this study.

This simulation setup builds a database for the investigation and shows that the process distributions at each time step changes are insignificant. Initially, we started calculating the histogram at each time step with identical binning for all of them. Afterwards, using the HD formulation, each histogram's distance to the other histograms (5999 other histograms) was calculated and stored in a matrix. Each row of this matrix shows the histogram difference at one time-step compared to the other ones. Therefore, this is a symmetric matrix with zeros on the diagonal. What is important is the maximum of all of the data in the matrix; in Figure 2.3a and 2.3b, we show the max of the HD at each time step for the aerodynamic model simulation outputs. The HD is a normalized metric, and the distances are shown in the percentage.

The plot shows a comparison of all the 18 million possible combinations to calculate the HD for each model output. For the simulation outputs (thrust and torque), the distributions' difference does not exceed 2.21%. This shows a sound coherence in the statistics at each time step. Therefore, we can conclude that building a surrogate model on a limited number of time steps, or even one time step, is enough, and it is not necessary to create a surrogate model on every time step as predicted by the aerodynamic model form.



(a) (b)

Figure 2.3: Maximum HD for thrust and torque at each time step. The upper and lower bounds for the *extreme* of the HD are indicated.

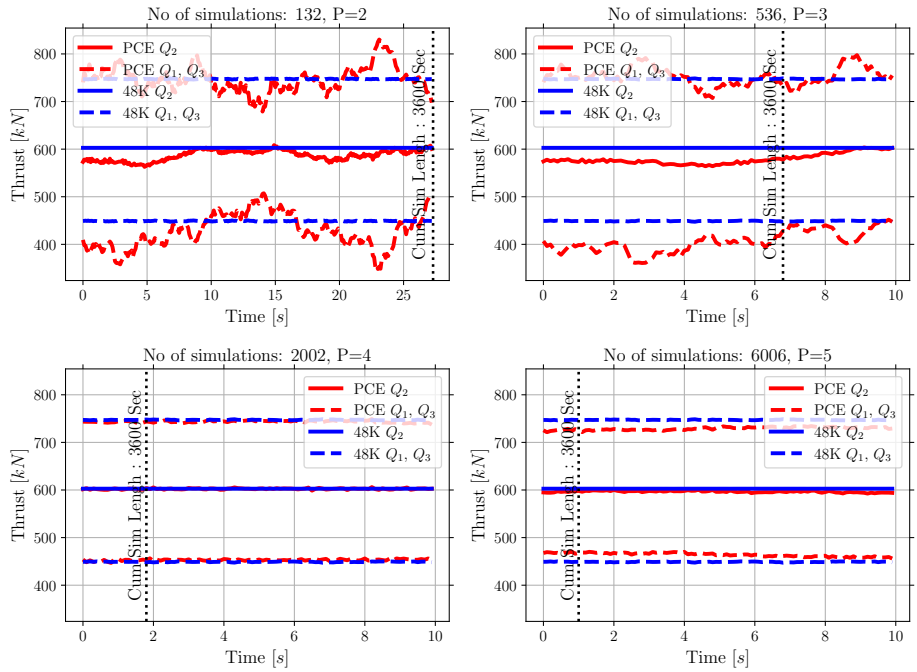
2.6.2 PCE Surrogate Model Construction

We use the same simulation setup as explained above for the reference case to run the specified number of simulations in Table 2.1. These simulations are input for building the surrogate models. The number of samples drawn from the 10 dimensional uniform random space is equal to the number of simulations in Table 2.1. The employed sampling method is Sobol, as tests show it provides a better convergence for the PCE.

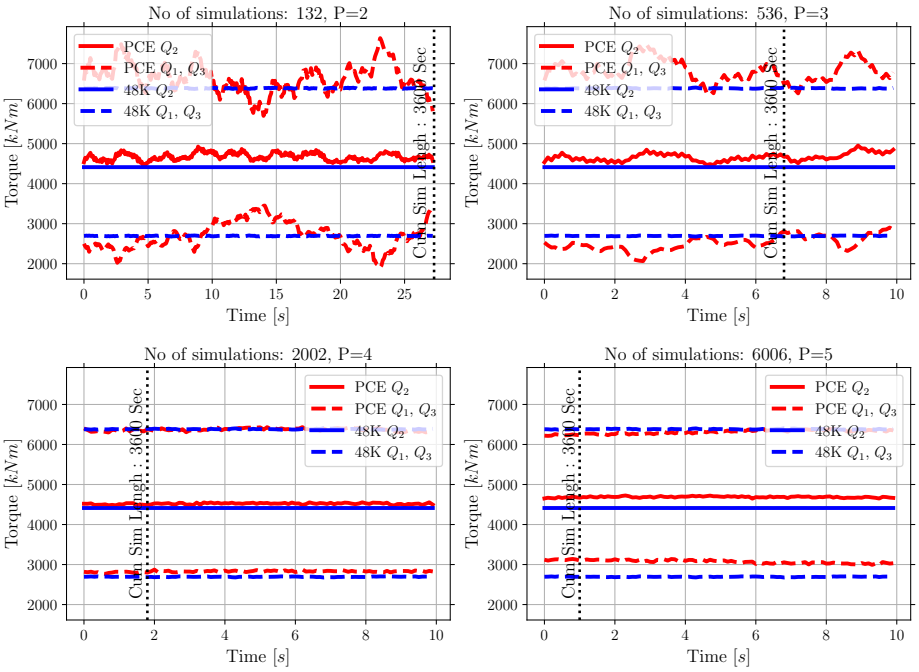
Referring to Figure 2.3, and the discussion in Section 2.6.1, the changes in statistical properties at each time step are minimal. Therefore, one surrogate model that can accurately emulate the sectional statistics of the aerodynamic simulations' output would suffice. Knowing this means building surrogate models is more feasible from a computational cost perspective.

As explained in Section 2.5, we fit surrogate models at every time step of a large set of short simulations instead of a few long ones for increasing polynomials order P. The number of simulations is based on the polynomial order as shown in Table 2.1. The length of the simulations that the surrogate model is built on every time step is the last column in Table 2.1 to keep the cumulative length of the simulations at $3600s$. Although it was unnecessary, for polynomial degrees 3 to 5, we build surrogate models at every time step of the whole $10s$ worth of simulations to have an acceptable sample size for direct comparison. The `chaospy` [42] toolbox was used to perform the task of building these surrogate models.

Figure 2.4 compares the descriptive statistics (first quartile Q_1, second quartile Q_2 and third quartile Q_3) for both thrust and torque from the reference case with 48K simulation outputs and 48K MCS of the surrogate model build at each time step. The results in Figure 2.4 show the PCE fits for four polynomial degrees; P on each plot indicate the polynomial degree. As the polynomial degree increases from $P = 2$ to $P = 4$, the fit to the reference case improves, as is expected. However, it seems there is more error in the mean value and quantiles when we move from $P = 4$ to $P = 5$ for both thrust and torque. This increase in the error is explained further in Section 2.6.3 and Figure 2.6. We calculate the average difference for the MCS and reference case over time for Q_1, Q_2 and Q_3 from Figure 2.4 for polynomials $P = 4$ and $P = 5$. This is presented in Table 2.2.



(a) Thrust



(b) Torque

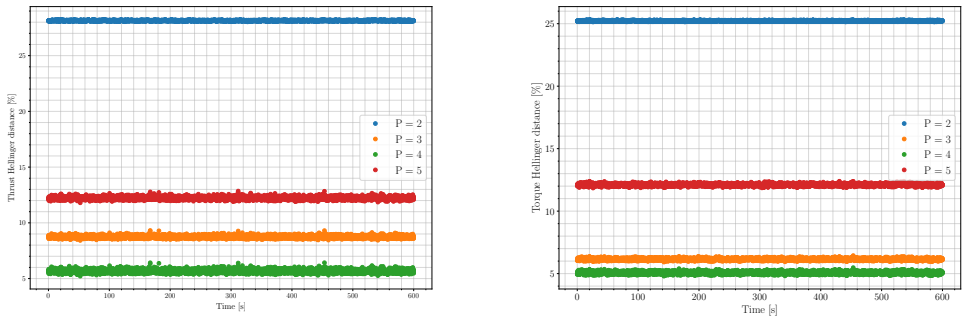
Figure 2.4: The Q_1, Q_2 and Q_3 comparison from the reference case ($48K$ simulations) and extracted values from PCEs for both trust and torque. The number of simulations used to build the PCEs and polynomial degree P are mentioned on the plots. The cumulative data length of $3600s$ sufficient to build surrogates is shown with the vertical line.

2.6.3 Surrogate Model MCS

PCE surrogate models were constructed in the previous section. Those surrogates can now be exercised via MCS to quickly provide output sampled statistics without actually running further simulations. We initially ranked the surrogate models constructed for each time-step based on their mean values and standard deviations for each polynomial order separately to choose a single PCE surrogate model for the MCS carried out next. As the surrogate models are constructed as PCEs, mean and standard deviation extraction is a simple step from the PCE coefficients [48]. After ranking the surrogate models, we selected the middle mean surrogate model from the ranked succession for each polynomial order. That provided us 4 surrogate models for thrust and 4 surrogate models for torque, one per polynomial order.

Next, we took the selected surrogate models thorough MCS of the surrogate models 10^6 times. Essentially, we took random samples from our 10 dimensional random domains 10^6 times and inserted those in the PCEs (Eq. (2.4) and Eq. (2.5)). The MCS outputs are then used to check the surrogate model's accuracy. One can argue that this method is cherry-picking the surrogate models. This argument is valid for the low order ($P = 2, 3$) polynomial surrogate models. However, from Figure 2.4 we know these polynomial are not accurate regardless of which one we choose. This inaccuracy is more visible in Figure 2.6. For polynomials of orders 4 and 5, referring to Figure 2.4, the polynomial selection procedure induces an insignificant effect on the statistics.

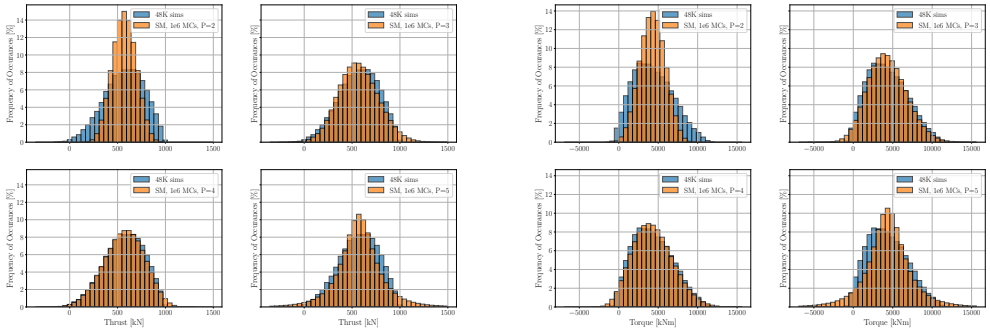
To verify the surrogate models' accuracy, we use the HD explained in section 2.5. This time, the HD shows the difference between the surrogate model's 10^6 MCS outputs per polynomial order with the reference case at every time step. This procedure provides a vector of HDs for each polynomial degree, where the vector's length is the same as the number of time steps in the simulations. As HD is sensitive to binning, the bins are identical for each polynomial order surrogate model. The same bins were used to calculate the reference simulations' histograms at each time step. Figure 2.5 shows the average HD changes within a narrow band for each polynomial degree. For the order 4 and 5 polynomilas, we calculate the average of the HDs over $600s$. The averaged HD presented in Table 2.2 serves as an error metric for the surrogate models.



(a) Thrust (b) Torque

Figure 2.5: HD between the different polynomial order surrogate models for one million MCS of the selected thrust and torque surrogates and the reference case simulations $48K$ at every time step

Afterwards, we compare the histogram of those with one arbitrary time step of the reference case of 48000 simulations. For each polynomial degree, regardless of the reference case time-step location in the time series, the difference between the reference case and the MCS only depends on the polynomial order. In other words, the difference between the MCS result histogram and the reference case histogram was only dependent on the polynomial degree and not the position of the time-step in the time series as expected for a stationary process. Figure 2.6 compares the histogram of 10^6 MCS for the middle mean ranked surrogate model to the reference case at one arbitrary time step for four polynomial degrees.



(a) Thrust (b) Torque

Figure 2.6: Thrust and torque Surrogate models with one million MCS vs the reference case histogram

Figure 2.6 shows how the surrogate models match the simulations output histogram. It is visible that the polynomials order $P = 3$ and $P = 4$ can cover the non-linearity in the results, while the second-order polynomial cannot. Polynomial order $P = 5$ does not work well for both thrust and torque. Although we met the rule of thumb for the number of simulations as mentioned in Table 2.1, this shows an *under fit* for $P = 5$. This means more simulations are required to make the fit feasible. Both Figures 2.5 and 2.6 show that $P = 4$ provides an acceptable accuracy for the surrogate models. Therefore, further tests on the surrogate models with $P = 5$ appear unnecessary. From Figures 2.5 and 2.6 we can conclude that the PCE surrogate model with polynomial order $P = 4$ is accurate enough to emulate the aerodynamic model with an acceptable accuracy, while covering its non-linearity.

Another metric to show the accuracy of the surrogate model is Normalized Root Mean Square Error (NRMSE), also known as L_2 norm error. The NRMSE is calculated for every time step for the first $10s$ by running MCS for the surrogate model $P = 4$ for the same input as the reference case simulations. This means we use the same samples we took from the $10 - D$ random variables to generate the unsteady wind and then calculate the $48K$ reference case to run the surrogate model MCS. Figure 2.7 shows the error against time. As expected and visible from Figure 2.4 the NRMSE is higher for torque and lower for thrust. In both cases, the maximum NRMSE is less than 10%. This error is deemed acceptable as the surrogate model aims to provide overall accurate statistics and not point-to-point accuracy in the estimation and is necessarily a trade-off between speed and accuracy in the intended applications. Recall from Equations (2.4) and (2.5) that the PCE method is formulated as an expansion over the space formed by polynomials which are functions of random variables. The simulation method of fitting the PCE coefficients is essentially performing a statistical fit across the summative set of simulation results, rather than optimizing the surrogate fit to a specific simulation. The NRMSE comparison here is therefore perhaps unfair to the intent of the PCE model, the earlier comparisons of MCS histograms and HDs more appropriate metrics for the proposed PCE surrogate model approach.

Figures 2.5, 2.6 and 2.7 show the PCE surrogate model has succeeded with the samples from the $10-D$ uniform distribution and converts them to an approximately Weibull distribution for both thrust and torque. This result highlights the ability of

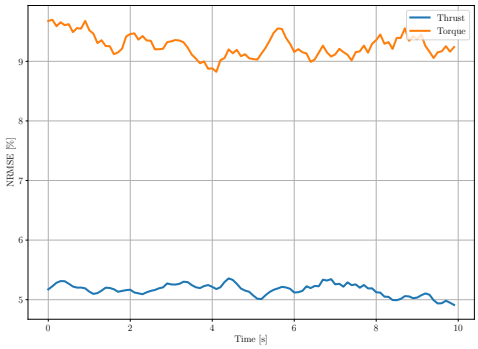


Figure 2.7: Surrogate models $P = 4$ NRMSE for both Thrust and Torque with respect to the reference case

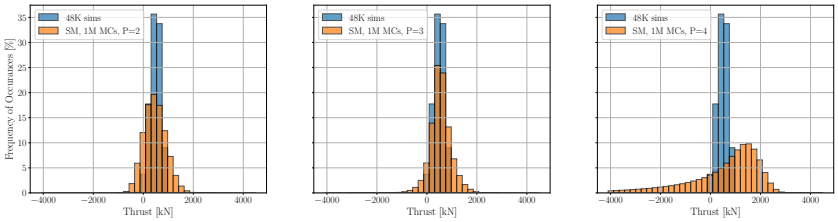
the PCE surrogate in this study to deal with the inherent non-linearity of the combination of unsteady wind generation and aerodynamic model together.

Table 2.2: The average HD for polynomials $P = 4$ and $P = 5$ and the quantile values difference for 1M MCS for thrust and torques with respect to the $48K$ reference case.

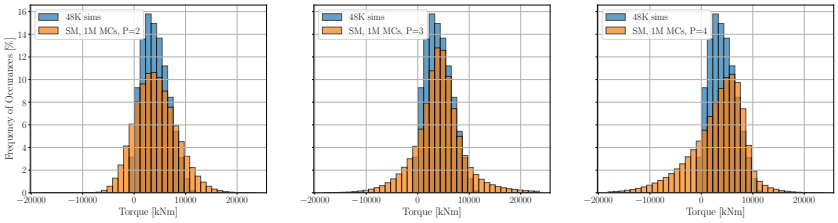
Case	No of Sim.	Avg.HD [%]		Thrust [%]			Torque [%]		
		Thrust	Torque	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3
PCE P = 4	2002	5.6	5.0	0.90	0.05	0.50	4.94	2.44	0.21
PCE P = 5	6006	8.7	6.1	3.36	1.03	2.42	14.12	6.22	1.10

2.6.4 SGQ PCE results

As mentioned in Section 2.5, we also calculated the coefficients for the PCE using the SGQ method for the polynomial orders 2, 3 and 4. We used the same procedure described in the previous sections to run the simulations, build the surrogate models (using `chaospy`), and select the surrogate models. This results in 6 surrogate models (3 for thrust and 3 for torque). We checked the accuracy of the surrogate models in the same manner as explained in the previous section. We ran 10^6 MCS on the 6 selected surrogate models and compared the histograms of the results with the histogram from one arbitrary time step from the reference case of 48000 simulations. The results of the investigation are presented in Figure 2.8.



(a) Thrust

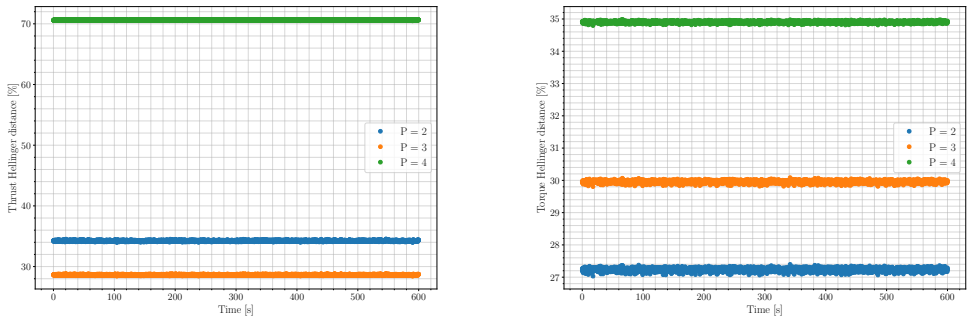


(b) Torque

Figure 2.8: Thrust and torque Surrogate models based on SGQ with one million MCS vs the reference case histogram

The results in Figure 2.8 show the SGQ method is less accurate than the point collocation method used in the previous section. Although the SGQ method requires more evaluation points than the point collocation method, the under-performance of the SGQ method is consistent for all the polynomial orders. As the initial accuracy test for the SGQ method did not provide comparable results with the point collocation method, we did not pursue further investigation of the SGQ method in this study. This finding is in line with literature that shows point collocation typically outperforms the SGQ in accuracy and efficiency [49].

The HD Figures 2.9a and 2.9b show the difference between the SGQ surrogate model's 10^6 MCS outputs per polynomial order with the reference case at every time step. The HD is much larger than what we showed in Figure 2.5 using the point collocation method to build the surrogate models.



(a) Thrust (b) Torque

Figure 2.9: HD between the different polynomial order surrogate models for one million MCS of the selected thrust and torque SGQ surrogates and the reference case simulations $48K$ at every time step

2.6.5 Surrogate model efficiency

Building the surrogate models aims to emulate the output of the actual model in an accurate and computationally efficient fashion. To inspect success in this respect, we start with computation time required to run 2002, 6006 and reference case $48K$, $600s$ aerodynamic simulations. The previous section shows that we do not need the $600s$ length simulations to build the surrogate models. Based on what we showed in Section 2.6.3, the required time to run $1s$ simulations (10-time steps) and building one surrogate model are provided. The computational time in Table 2.3 includes the unsteady wind generation. Additionally, we record the time required to build one surrogate model. The time to build one surrogate model for both thrust and torque is similar, and the average is provided here. As IEC standards [4] asks for at least six aerodynamic simulations per average wind speed, we register the time for that set of aerodynamic simulations also. We perform aerodynamic simulations and surrogate model building on Compute Canada WestGrid clusters. The CPU time for the aerodynamic simulations and building the surrogate models is presented in Table 2.3.

The computational time to build one surrogate model is long for $P = 5$. This is due to employing the point collocation method to calculate the PCEs coefficients. The point collocation method is inherently a regression method, using least squares to minimize the error [42]. For a more complex aeroelastic model, the simulation times would be increased, potentially substantially, shifting the balance of compu-

Table 2.3: The computational time required to run aerodynamic simulations, building the surrogate models. SM stands for Surrogate Model, and CP stands for Common Practice.

Case	No of Sim.	Sim. length	Comp. time	1 SM
PCE P = 4	2002	1s	10s	1min 2s
PCE P = 5	6006	1s	31s	15min 54s
CP	6	600s	18.62s	N/A
Ref. case	48000	600s	41h 55min 48s	N/A

tational time from PCE construction toward aeroelastic simulations. Of course, the aeroelastic simulations may be parallelized on available computing infrastructure.

After building the surrogate models, we ran large sets of MCS for the PCE surrogate models with the polynomial order 4 (as it is the most accurate one) and tracked the required time to run the MCS. All the MCS were performed on Compute Canada WestGrid clusters. The computation time for the MCS is shown in Table 2.4. As the computational time difference between thrust and torque is insignificant, the one which took longer is mentioned here.

Table 2.4: Computational time to run MCS on the surrogate models with polynomial order 4.

No. MCS simulations	Surrogate model $P = 4$
10, 100, 1K	$<$ 0.1s
10 K	$<$ 0.25s
48 K	0.85s
100 K	1.70s
1 M	30.84s
10 M	5min 51s
100 M	55min 33s
288 M	2h 40min 1s
500 M	4h 46min 36s

The number of time steps in the reference case is 288 M (6000 time-steps multiplied by 48000 aerodynamic simulations). Therefore, to have a fair comparison we can compare computational time for the reference case in Table 2.3 with 288 M MCS in Table 2.4. Adding up the computational time required for the surrogate model

input simulations and building the surrogate model, still, the MCS is more efficient by a big margin.

The ease of running MCS provides the ability to have more samples from the random domain. More samples from the random domain cover more of the statistical domain and capture the extreme loads more efficiently than running time marching aerodynamic simulations and extrapolation. Figures 2.10 and 2.11 present the comparison between different MCS set sizes and the reference case aerodynamic simulations, maximum, 99th percentile P_{99}, 95th percentile P_{95} and 90th percentile P_{90}. The maximum load and the percentiles extracted from the $P = 4$ surrogate models run for both thrust and torque are shown in relation to the number of MCS.

According to IEC standards [4], the maximum load needs to be calculated based on the mean of the max (mean-max) of at least six seeds of unsteady wind aerodynamic simulations per average wind speed. Therefore, we randomly grouped the reference case simulations (48K simulations) into 8000 unique groups to have a fair comparison with the common practice. Afterwards, the mean of the max per group, and the 90th percentile P_{99}, 95th percentile P_{95} and 90th percentile P_{90} of each group is calculated. These data are presented in Figures 2.10 and 2.11 as clouds of dots for both thrust and torque.

To extract the maximum value that matches the full reference case aerodynamic simulations, we need to run a large number of MCS. Figure 2.10 and 2.11 show for the PCE order 4 the maximum thrust and torque from the MCS matches the reference case around almost 1000 MCS. Looking at P_{99}, P_{95} and P_{90} for both surrogate models, the convergence happen at around the same number of MCS. Figures 2.10 and 2.11 show after $1K$ or $100K$ MCS the percentiles are close to the reference case. Interestingly, the mean-max output from the grouped aerodynamic simulations has a wide distribution. This shows the inaccuracy of looking at a small number of simulations to calculate the extreme loads. This distribution is smaller for the percentile data; however, it is not comparable with the convergence of MCS outputs. Also, looking at the grouped simulations output, and compared to the standard practice (mean-max) and P_{99}, illustrates the conservative design approach of the IEC standards [4].

Referring to the computational time required to build the surrogate model and

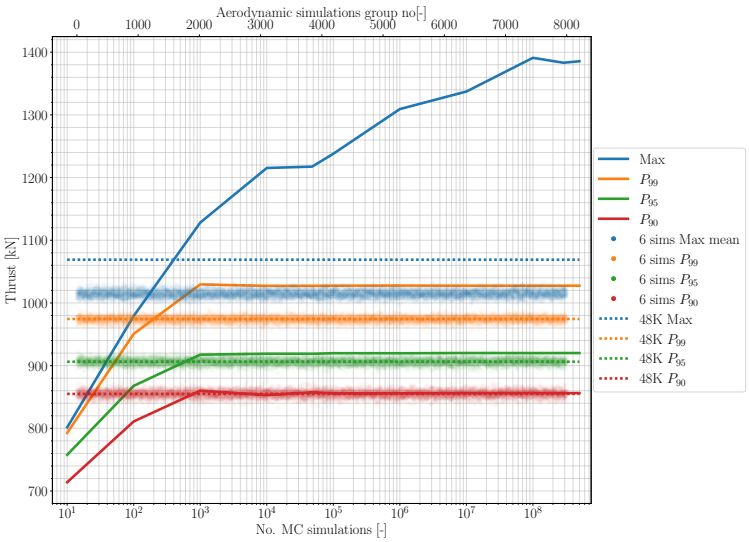


Figure 2.10: Maximum load and percentile comparison between the aerodynamic simulations reference case, the MCS and groups aerodynamic simulations for thrust

then running the MCS, these plots show promising results to extract accurate extreme loads from the surrogate models in a computationally efficient manner. Here again, it is emphasized that the utility of the proposed PCE-based surrogate, with MCS of the constructed surrogate and examination of the statistical loads distributions is the key contribution of the work. The point-to-point accuracy of the model for a single run of the surrogate, as discussed earlier is not the focus of the surrogate, but rather the overall computational costs and accuracy in spread of loading conditions that is our focus.

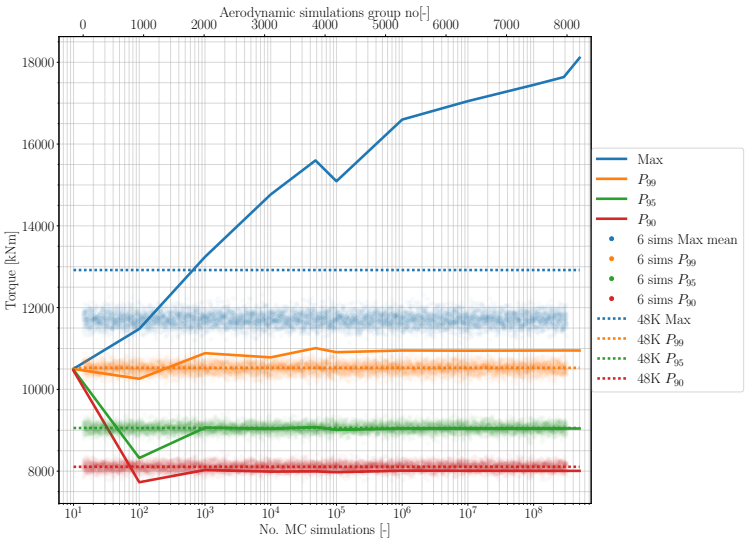


Figure 2.11: Maximum load and percentile comparison between the aerodynamic simulations reference case, the MCS and groups aerodynamic simulations for torque

2.7 Conclusion

This paper's aim is to build a non-intrusive surrogate model of time marching aerodynamic simulations. The form of the surrogate model used in this paper is a PCE. In Section 2.5, we explained the aerodynamic model used for this study. Then, we briefly described the method that we are using to build the PCEs. One major challenge with the building of the surrogate models is the *curse of dimensionality*, which we tried to tackle by using a reduced Veers model.

We showed how by increasing the number of simulations, the results' statistics converge and do not change in time. As a result, building a few accurate surrogate models, or even one, for a small length of time would suffice for our purpose. In other words, time does not have any meaning in the sectional statistics. Therefore, to build an accurate surrogate model, we can significantly reduce the simulation length while increasing the number of simulations. In the results section, we showed the surrogate model using a fourth-order polynomial built on 2002 simulations with a length of $2s$ gives us sufficiently accurate results in large MC runs to obtain output statistics. Afterwards, we demonstrate the surrogate model's efficiency by comparing the computational time required to run the aerodynamic simulations and build the surrogate model with the required time for running MCS to have accurate statistics.

Also, we show the high percentile values extracted from the MCS match the reference case with a relatively low number of MCS and thus computationally efficient.

The BEM-based aerodynamic model approach is well known in the literature and research. We choose $30s$ transient time for the simulations to ensure they do not include any transition results. As the model is a less complex BEM which is quick to run, this is not a challenge. However, for future work with actual aeroelastic codes (e.g. FAST), a smart way to deal with initialization time is essential; otherwise, increasing the number of simulations and the model complexity would be very expensive. For example, if the required initialization time is $60s$ (default in FAST), and we want to increase the number of simulations from 6 six hundred second simulations (minimum requirement according to [4]) to 6006 two-second simulations, we are not doing any good in terms of computational cost. Aeroelastic and longer wakes will be studied for this challenge and blended or common initialization period will be trialled.

Another challenge is the practical application of this surrogate model. The surrogate model that we build in this study is one or a few time steps, each inherently the same due to the statistical similarity. If we want to build one time series from this surrogate model, we have to sample the 10 dimensional random domain for the number of time steps to have a time series to post-process. For example, suppose we want to have a $600s$ time series of thrust or torque with the time step of $0.1s$ for the aerodynamic model that we developed in this study. In that case, we need to take 6000 samples from the 10 dimensional uniform distribution random domain, and run MCS for each. However, this would provide us with 6000 thrust and torque values, which will miss the auto-correlation, which is inherent in the generated unsteady wind, in the results. This drawback is crucial if we, for instance, want to calculate fatigue loads from the surrogate model. This challenge will require a surrogate form capable of resolving the correlation between time steps. [18] did this previously for intrusive PCEs of an aerodynamic model, however as mentioned before, that can get very complicated for more advanced models.

Using non-conventional polynomials, such as what [18] did, might result in a more efficient polynomial that requires fewer number simulations to build the surrogate model. Finally, we want to implement the method on commercial wind turbine simulation packages such as FAST to test the approach in future work. It is important to

again note however that the physics model used in the current work is equivalent to FAST, just conveniently formulated in Python for our surrogate model development efforts. The notion of the "reduced Veers model" worked for the aerodynamic simulations we used in this study. However, this reduced model would not be efficient and sufficient to move towards commercial wind turbine packages. Therefore, a new approach to reduce the data in a "turbulence box" [24] spatially and temporally would be necessary. Similar work has been done in [50] and will be explored together with expansions of the methods invested by [22] using velocity increments across the wind field.

Code availability

The code and data that support the findings of this study are available from the corresponding author, RH, upon request.

Competing interests

The authors confirm there are no competing interests present.

Acknowledgements

We greatly acknowledge the funding for this study by the Natural Sciences and Engineering Research Council of Canada (NSERC). This research was enabled in part by support provided by WestGrid (www.westgrid.ca) and Compute Canada Canada (www.computecanada.ca).

Bibliography

[1] Jason M Jonkman, Marshall L Buhl Jr, et al. Fast user's guide. 2005.

[2] EA Bossanyi. Gh bladed theory manual. *GH & Partners Ltd*, 2:56–58, 2003.

[3] Torben J Larsen and Anders Melchior Hansen. How 2 hawc2, the user's manual. 2007.

[4] Wind energy generation systems - part 1: Design requirements. Standard, International Electrotechnical Commission, 2019.

[5] Arne van Garrel. Development of a wind turbine aerodynamics simulation module. Technical report, Energy research Centre of the Netherlands, 01 2003.

[6] Bruno Sudret. *Uncertainty propagation and sensitivity analysis in mechanical models–Contributions to structural reliability and stochastic spectral methods.* PhD thesis, 2007.

[7] N. Dimitrov, M. Kelly, A. Vignaroli, and J. Berg. From wind to loads: wind turbine site-specific load estimation using databases with high-fidelity load simulations. *Wind Energy Science Discussions*, 2018:1–39, 2018. doi: 10.5194/wes-2018-18. URL `https://www.wind-energ-sci-discuss.net/wes-2018-18/`.

[8] Laura Schröder, Nikolay Krasimirov Dimitrov, David Robert Verelst, and John Aasted Sørensen. Wind turbine site-specific load estimation using artificial neural networks calibrated by means of high-fidelity load simulations. In *Journal of Physics: Conference Series*, volume 1037, page 062027. IOP Publishing, 2018.

[9] LMM van den Bos, B Sanderse, L Blonk, WAAM Bierbooms, and GJW van Bussel. Efficient ultimate load estimation for offshore wind turbines using in-

terpolating surrogate models. In *Journal of Physics: Conference Series*, volume 1037, page 062017. IOP Publishing, 2018.

[10] Nikolay Dimitrov. Surrogate models for parameterized representation of wake-induced loads in wind farms. *Wind Energy*, 22(10):1371–1389, 2019.

[11] Turaj Ashuri, Tao Zhang, Dong Qian, and Mario Rotea. Uncertainty quantification of the levelized cost of energy for a 20 MW research wind turbine model. In *34th Wind Energy Symposium*, page 1998, 2016.

[12] Juan Pablo Murcia, Pierre-Elouan Réthoré, Nikolay Dimitrov, Anand Natarajan, John Dalsgaard Sørensen, Peter Graf, and Taeseong Kim. Uncertainty propagation through an aeroelastic wind turbine model using polynomial surrogates. *Renewable Energy*, 119:910–922, 2018.

[13] Laura Schröder, Nikolay Krasimirov Dimitrov, and John Aasted Sørensen. Uncertainty propagation and sensitivity analysis of an artificial neural network used as wind turbine load surrogate model. In *Journal of Physics: Conference Series*, volume 1618, page 042040. IOP Publishing, 2020.

[14] René MM Slot, John D Sørensen, Bruno Sudret, Lasse Svenningsen, and Morten L Thøgersen. Surrogate model uncertainty in wind turbine reliability assessment. *Renewable Energy*, 151:1150–1162, 2020.

[15] Laura Schröder, Nikolay Krasimirov Dimitrov, and David Robert Verelst. A surrogate model approach for associating wind farm load variations with turbine failures. *Wind Energy Science*, 5(3):1007–1022, 2020.

[16] Haipeng Wang, Xiao Jiang, Yun Chao, Qian Li, Mingzhou Li, Tao Chen, and Weirui Ouyang. Numerical optimization of horizontal-axis wind turbine blades with surrogate model. *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, page 0957650920976743, 2020.

[17] Thanasis Barlas, Néstor Ramos-García, Georg Raimund Pirrung, and Sergio González Horcas. Surrogate-based aeroelastic design optimization of tip extensions on a modern 10 mw wind turbine. *Wind Energy Science*, 6(2):491–504, 2021.

[18] Manuel Fluck and Curran Crawford. A fast stochastic solution method for the blade element momentum equations for long-term load assessment. *Wind Energy*, 21(2):115–128, 2018.

[19] Manuel Fluck and Curran Crawford. Fast analysis of unsteady wing aerodynamics via stochastic models. *AIAA Journal*, 55(3):719–728, 2016.

[20] Paul S Veers. Three-dimensional wind simulation. Technical report, Sandia National Labs., Albuquerque, NM (USA), 1988.

[21] Manuel Fluck and Curran Crawford. A stochastic aerodynamic model for stationary blades in unsteady 3d wind fields. In *Journal of Physics: Conference Series*, volume 753, page 082009. IOP Publishing, 2016.

[22] M. Fluck and C. Crawford. An engineering model for 3-d turbulent wind inflow based on a limited set of random variables. *Wind Energy Science*, 2(2):507–520, 2017. doi: 10.5194/wes-2-507-2017. URL `https://www.wind-energ-sci.net/2/507/2017/`.

[23] Ervin Bossanyi, Tony Burton, D Sharpe, and N Jenkins. *Wind energy handbook*. New York: Wiley, 2011.

[24] Bonnie J Jonkman. Turbsim user's guide: Version 1.50. 2009.

[25] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, 1992. doi: 10.1137/1.9781611970081.

[26] Il'ya Meerovich Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4):784–802, 1967.

[27] Sergei Kucherenko, Daniel Albrecht, and Andrea Saltelli. Exploring multidimensional spaces: a comparison of latin hypercube and quasi monte carlo sampling techniques. 05 2015.

[28] Rick Lupton. bemused. `hhttps://github.com/ricklupton/bemused`, 2019.

[29] Jason Jonkman, Sandy Butterfield, Walter Musial, and George Scott. Definition of a 5-mw reference wind turbine for offshore system development. Technical

report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2009.

[30] RC Lupton. *Frequency-domain modelling of floating wind turbines*. PhD thesis, University of Cambridge, 2015.

[31] A. Basu, H. Shioya, and C. Park. *Statistical Inference: The Minimum Distance Approach*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. CRC Press, 2011. ISBN 9781420099669.

[32] Ernst Hellinger. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1909(136):210–271, 1909.

[33] Sun Hye Kim and Fani Boukouvala. Machine learning-based surrogate modeling for data-driven optimization: a comparison of subset selection for regression techniques. *Optimization Letters*, pages 1–22, 2019.

[34] Roger G Ghanem and Pol D Spanos. *Stochastic finite elements: a spectral approach*. Courier Corporation, 2003.

[35] Dongbin Xiu and George Em Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *Journal of computational physics*, 187 (1):137–167, 2003.

[36] Stephen Tyson, Diane Donovan, Bevan Thompson, Steve Lynch, and Marvin Tas. Uncertainty modelling with polynomial chaos expansion: Stage 1 - final report, 2015.

[37] Arun Kaintura, Tom Dhaene, and Domenico Spina. Review of polynomial chaos-based methods for uncertainty quantification in modern integrated circuits. *Electronics*, 7(3):30, 2018.

[38] Bruno Sudret. Polynomial chaos expansions and stochastic finite element methods. *Risk and reliability in geotechnical engineering*, pages 265–300, 2015.

[39] Dongbin Xiu and George Em Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM journal on scientific computing*, 24 (2):619–644, 2002.

[40] Roger Ghanem, David Higdon, and Houman Owhadi. *Handbook of uncertainty quantification*. Springer, 2017.

[41] Michael Eldred, Clayton Webster, and Paul Constantine. Evaluation of non-intrusive approaches for wiener-askey generalized polynomial chaos. In *10th AIAA Non-Deterministic Approaches Conference*, page 1892, 2008.

[42] Jonathan Feinberg and Hans Petter Langtangen. Chaospy: An open source tool for designing methods of uncertainty quantification. *Journal of Computational Science*, 11:46–57, 2015.

[43] Dongbin Xiu, Didier Lucor, C-H Su, and George Em Karniadakis. Stochastic modeling of flow-structure interactions using generalized polynomial chaos. *Journal of Fluids Engineering*, 124(1):51–59, 2002.

[44] Serhat Hosder, Robert Walters, and Michael Balch. Efficient sampling for non-intrusive polynomial chaos applications with multiple uncertain input variables. In *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1939, 2007.

[45] Olivier Le Maître and Omar M Knio. *Spectral methods for uncertainty quantification: with applications to computational fluid dynamics*. Springer Science & Business Media, 2010.

[46] Sergey Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Soviet Math. Dokl.*, volume 4, pages 240–243, 1963.

[47] Suhasini Subba Rao. A course in time series analysis. 2008.

[48] Nicola E Owen, Peter Challenor, Prathyush P Menon, and Samir Bennani. Comparison of surrogate-based uncertainty quantification methods for computationally expensive simulators. *SIAM/ASA Journal on Uncertainty Quantification*, 5 (1):403–435, 2017.

[49] Michael Eldred and John Burkardt. Comparison of non-intrusive polynomial chaos and stochastic collocation methods for uncertainty quantification. In *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, page 976, 2009.

[50] Q Guo and B Ganapathysubramanian. Incorporating a stochastic data-driven inflow model for uncertainty quantification of wind turbine performance. *Wind Energy*, 20(9):1551–1567, 2017.

Chapter 3

Data-driven surrogate model for wind turbine damage equivalent load

This chapter is submitted to the Wind Energy Science journal and accepted for review as:

Haghi, R. and Crawford, C.: Data-driven surrogate model for wind turbine damage equivalent load, Wind Energ. Sci. Discuss. [preprint], https://doi.org/10.5194/wes-2023-157, in review, 2023.

Author Contribution

RH developed the necessary computer code and wrote the paper in consultation with and under the supervision of CC.

Abstract

Aeroelastic simulations are used to assess wind turbines in accordance with IEC standards in the time domain. Doing so can calculate fatigue and extreme loads on the wind turbine's components. These simulations are conducted for several reasons, such as reducing safety margins in wind turbine component design by covering a wide range of uncertainties in wind and wave conditions, and meeting the requirements of the digital twin, which needs a thorough set of simulations for calibration. Thus,

it's essential to develop computationally efficient yet accurate models that can replace costly aeroelastic simulations and data processing. We suggest a data-driven approach to build surrogate models for the DEL based on aeroelastic simulation outputs to tackle this challenge. Our method provides a quick and efficient way to calculate DEL using wind input signals without the need of time-consuming aeroelastic simulations. Our study will focus on utilizing a sequential ML method to map wind speed time series to DEL. Furthermore, we demonstrate the versatility of the developed and trained surrogate models by testing them for a wind turbine in the wake and using transfer learning to enhance their prediction.

3.1 Introduction

For design, optimization and maintenance purposes of a wind turbine, wind turbine researchers and engineers need to simulate a wind turbine's dynamic behaviour. This has been done based on the IEC standards [1] using time-marching aeroelastic codes such as FAST [2], HAWC [3] or Bladed [4]. We utilize these time-marching simulations to calculate extreme and fatigue loads on wind turbine components caused by wind and wave as the inputs. The time-marching simulations are necessary for our work and research as they enable us to consider the inherent and necessary non-linearity in the wind turbine models. As both wind and waves are stochastic processes, a large set of simulations is preferred to understand the turbine behaviour fully and consider the uncertainty the stochasticity introduces. However, this increase in the number of simulations increases the computational costs. One solution to this is developing a computationally efficient SM which is cheaper to run yet accurate for our purposes.

The concept of the SM can be traced back to the field of UQ analysis, as stated in [5]. SMs, emulators or response surfaces are simple representations of a complex model, which can map the input to the output. At the same time, they can encapsulate the complexity of the original model [6]. Asher et al. provides an overview of the different categories of SMs [7]. There are different methods to develop a SM such as PCE [8, 9], or GPR [10, 11]. Recently, the application of ANN and ML has become increasingly prevalent among researchers and engineers developing SMs [12, 13, 14, 15]. This trend can be attributed to the widespread recognition of the ANN as a way to approximate any complex function with a few layers with high accuracy [16], and the increase in the data accessibility and availability.

Researchers and engineers have been using SMs for increasingly diverse applications in the wind energy domain. In the load emulation domain, [17, 18, 19] utilized PCE, Kriging, and ANN SMs to approximate wind turbine loads by considering stochastic variables such as turbulence intensity, mean wind, and wind direction. [20] employed a GPR-based SM to predict the fatigue load on a wind turbine affected by the wake in an onshore wind farm. Similarly, [21] used multiple SMs, such as GPR and ANN, to map inflow parameters in an array of wind turbines to the fatigue loads of the wind turbines in that array. [22] used a GPR-based SM for UQ of an offshore wind turbine's fatigue based on a wide range of environmental and structural variables. [23] employed polynomial interpolation as an SM for estimating ultimate loads on a wind turbine, while [24] used a random forest-based SM for ultimate load emulation. [25] implemented a probabilistic SM for offshore wind turbine loads using chained GPR. [26] utilized an SM as an aerodynamic emulator for real-time testing of floating wind turbines. In a different approach, Fluck and Crawford used *intrusive* PCE to build a surrogate model for lifting line and BEM models [28, 27]. Similarly, Haghi and Crawford built SMs on a BEM model of NREL 5MW turbine simulations output time steps using *non-intrusive* PCE [29]. In their work, the SMs mapped the random phases in the unsteady wind generation [27, 30] to the output loads of the simulations at each time step.

As wind and waves are both uncertain, the high computational cost associated with the simulator in a DT may make it impractical to propagate uncertainty. Hence, employing a SM within the DT framework becomes beneficial when simulations are computationally expensive [31]. Also, using a surrogate model in a DT system creates the potential for the surrogate model to operate in real-time [32]. In recent years, DT for wind turbines has gained popularity among researchers and engineers. DTs have been used at different levels in the energy systems and wind turbine industries. Song et al. provided an overview of DT applications and challenges for energy systems in the future [33]. De Kooning et al. laid out an overview of DT applications in wind energy conversion [34]. Fahim et al. provided a method to develop a DT for wind turbines in a wind farm-level system using machine learning methods [35]. More specifically, with regard to DT applications for loads, Song et al. used measurements from the Block Island offshore wind farm to develop a DT for the turbines in the field [36]. In other work, Branlard et al. built a DT based on a linearized model of a wind

turbine [37]. Later, Branlard et al. developed a DT based on the Tetra spar floating platform full-scale prototype successfully [38]. With numerous instances of successful applications of DTs in the wind energy sector and the potential enhancements that a SM could bring to the DT framework, it is crucial to conduct further research on developing accurate and efficient SMs for wind energy systems.

Recently, there has been a surge in using ML and ANN techniques to create wind system SMs. This subject has garnered considerable attention and interest among professionals in the field. A recent study conducted by Schröder et al. utilized TL and physics-informed ML to enhance wind farm monitoring from SCADA data. The study aimed to improve the efficiency and effectiveness of wind farm monitoring using TL. The results showed that integrating TL and physics-informed ML can enhance the accuracy and reliability of wind farm monitoring systems [39]. Schröder et al. also used an ANN to build a SM that examined how changes in loads within a wind farm affect the reliability of wind turbine components. Their study aimed to evaluate the impact of load changes on wind turbine components' overall performance and reliability. The results showed that ANN-based SMs can provide valuable insights into the behaviour of wind turbine components under different load conditions [40]. Additionally, Mylonas et al. used a conditional variational autoencoder to create a probabilistic model of fatigue using SCADA data. Their goal was to predict the probability of fatigue load in wind turbine components using SCADA data. The results showed that ML-based methods predict fatigue accurately [41]. Lastly, Dimitrov and Göçmen used a time-based ML model LSTM to develop a virtual sensor that can predict and forecast the high-resolution load-time series of wind turbine components based on a series of environmental and turbine behaviour variable inputs. The results showed that ML-based time series models are accurate in their prediction and forecasting; however, a less complex ANN can still effectively predict outcomes [42].

3.1.1 Objective

The available literature and research indicate a lack of sufficient exploration and demonstration of a SM capable of mapping high-resolution environmental time series, specifically wind and/or wave for both on- and off-shore wind turbines, to the fatigue and extreme loads on wind turbine components. The development of such

a SM could potentially enable the prediction of the DEL of the wind turbine components using just a few input time series, thereby enhancing the efficiency of wind turbine control systems and increasing the overall lifespan of the turbine. Moreover, the use of this system in a DT framework would further enhance efficiency and facilitate real-time application.

Our ultimate is to develop a fully generalized SM that can predict wind turbine fatigue and extreme loads in any condition without the need for extra customization or tweaking based on wind, wake, and wave time history. This manuscript specifically begins to explore the approach by using sequential ML methods to build such a SM, which will map synthetic wind and wake time series to DEL. The objectives of the present manuscript are as follows:

- Building extensive wind time histories and wind turbine loads databases based on a QMC sampling of the synthetic wind generation input variables.
- Developing simple FCNN base SMs [43] that maps synthetic wind generation inputs to DEL [19], serving as a literature benchmark for performance and accuracy.
- Developing a sequential ML base SM using TCN [44] to project synthetic unsteady wind time series to wind turbine components DEL.
- Showing the capability of the *trained* sequential ML SMs by developing a TL framework to predict DEL while dealing with wake-induced synthetic wind time series.

3.1.2 Paper outline

This paper is organized as follows. Section 3.2 provides an overview of the methodology used in this study. The basics behind the data-driven models are then described in Sections 3.2.1 and 3.2.2. Sections 3.2.3 and 3.2.4 explain the process of building the databases in detail. Section 3.3, we delve into the essential prerequisites for constructing the databases, imparting knowledge to the SMs, and leveraging their predictive prowess for both the free stream and downstream wake. In the same section, we also compare the accuracy of different SM architectures developed in this study and discuss the amount of data required for training, as well as the limitations

of the developed SMs. The paper concludes in Section 3.4, where we summarize the main findings of this work and suggest future research in the area of wind turbine surrogates using sequential ML models.

3.2 Methodology

The presentation of the methodology section in this document has been adapted from the approach outlined in [40] due to its clarity and relevance to the current topic. The chosen framework is deemed to be an appropriate and effective means of conveying the necessary information in a concise and organized manner. The methodology used in this manuscript to map synthetic wind high-resolution time series to DEL is shown in Figure 3.1. It involves developing a sequential ML model combined with a FCNN architecture as the main SMs and utilizing a simpler FCNN for comparison purposes.

The configuration presented in Figure 3.1 has three blocks. The bottom block is for *Data generation*, which shows the procedure for building a database for the DEL from the input variables. The top two blocks are two methods to build a SM from the generated data and input variables. The middle block presents the approach to building a SM that maps high-resolution wind time series to DEL based on TCN-FCNN architecture. The top block exhibits the process of creating a FCNN that projects the input variables to DEL [19, 18]. The larger frameworks and three blocks can be segmented into twelve smaller stages. Each step is summarized below. Throughout this document, when we mention wind, we are specifically referring to unsteady wind.

(1) Specify the input variables space, their distributions and boundaries, and afterwards, generate n samples $\mathbf{X}$ from the predefined variables. To enable tracking, every sample has been indexed. The database is split into two for training and testing:

   (1a) *Training Input Variables* which includes 90% of the samples randomly selected. Therefore, the size of this database is $0.9n$. The indices of the randomly selected samples idx_{input} have been stored.

   (1b) *Testing Input Variables* which includes the 10% remaining of the samples. As a result, the size of this database is $0.1n$

(2) The n generated samples are the input to a wind generator. Each sample from the input variable space generates one synthetic wind time series with the length of t time steps.

(3) The n synthetic wind time series are stored in *Wind Database*. The database size is $n \times t$ where t is the number of time steps in the time series. For training/testing purposes, this database is split into two parts:

(3a) *Training Wind Database*, which includes 90% of the main synthetic wind time series database randomly selected. Consequently, the size of this database is $0.9n \times t$. The indices of the randomly selected samples idx_{wind} have been stored.

(3b) *Testing Wind Database*, which includes the remaining 10% of the main synthetic wind time series database. The size of this database is $0.1n \times t$.

(4) The *Wind Turbine Model* is an input to the *Aero-servo-elastic Simulator*. The model comprises three modules: aerodynamic, controller, and aeroelastic.

(5) *Aero-servo-elastic Simulator* is a time-marching solver that takes synthetic wind time series and wind turbine model as the input and delivers forces and moments, *loads*, time series at l wind turbine components as the output.

(6) All the n outputs of the previous step simulations are stored in a database. In the *Simulation Database*, each simulation includes the l wind turbine components load time series for t time steps for one sample from the input variable space. Therefore, the database size is $n \times m \times t$

(7) The time-series output is analyzed to determine the DEL of the loads on the l wind turbine components.

(8) For every wind turbine component in the *DEL database*, each simulation output yields a single DEL data point. Therefore, the database size is $n \times l$. Every row in the DEL database has an index that corresponds to the index of its input variable sample. As we train two SMs with the database, we split the database into training and testing databases twice. Thus, there appears to be an overlap between the testing and training databases. However, as we have utilized them to train and test two distinct SMs, we do not anticipate any issues arising from this situation.

(8a) *Training DEL Database* members are selected based on the idx_{input} indices. Therefore, this database includes 90% of the DEL and the size is $0.9n \times l$

(8b) *Testing DEL Database* which includes the remaining 10% members of the DEL databases. Hence, this database size is $0.1n \times l$

(8c) *Training DEL Database* members are selected based on the idx_{wind} indices. Therefore, this database includes 90% of the DEL, and the size is $0.9n \times l$. As mentioned before, there is an overlap between this database and the database in 8a.

(8d) *Testing DEL Database* which includes the remaining 10% members of the DEL databases. Correspondingly, this database size is $0.1n \times l$

(9) The SM with FCNN composition trains and validates using the databases in 1a as the input and 8a as the output.

(9a) For testing, the trained FCNN SM takes the database in 1b as the input and provides *FCNN Prediction DEL* as the output.

(10) The SM with TCN-FCNN architecture trains and validates using the databases in 3a as the input and 8c as the output.

(10a) For testing, the trained TCN-FCNN SM takes the database in 3b as the input and provides *TCN Prediction DEL* as the output.

(11) By comparing 9a with 8b, one can determine the accuracy of the of the trained FCNN SM.

(12) By comparing 10a with 8d, one can determine the accuracy of the of the trained TCN-FCNN SM.

The aim to build and train a simple FCNN SM is to compare the accuracy and performance of TCN-FCNN SM to it. The FCNN SM is not the ground truth in this piece of work; however, it has proven to provide acceptable accuracy for the similar input variable space [19, 18].

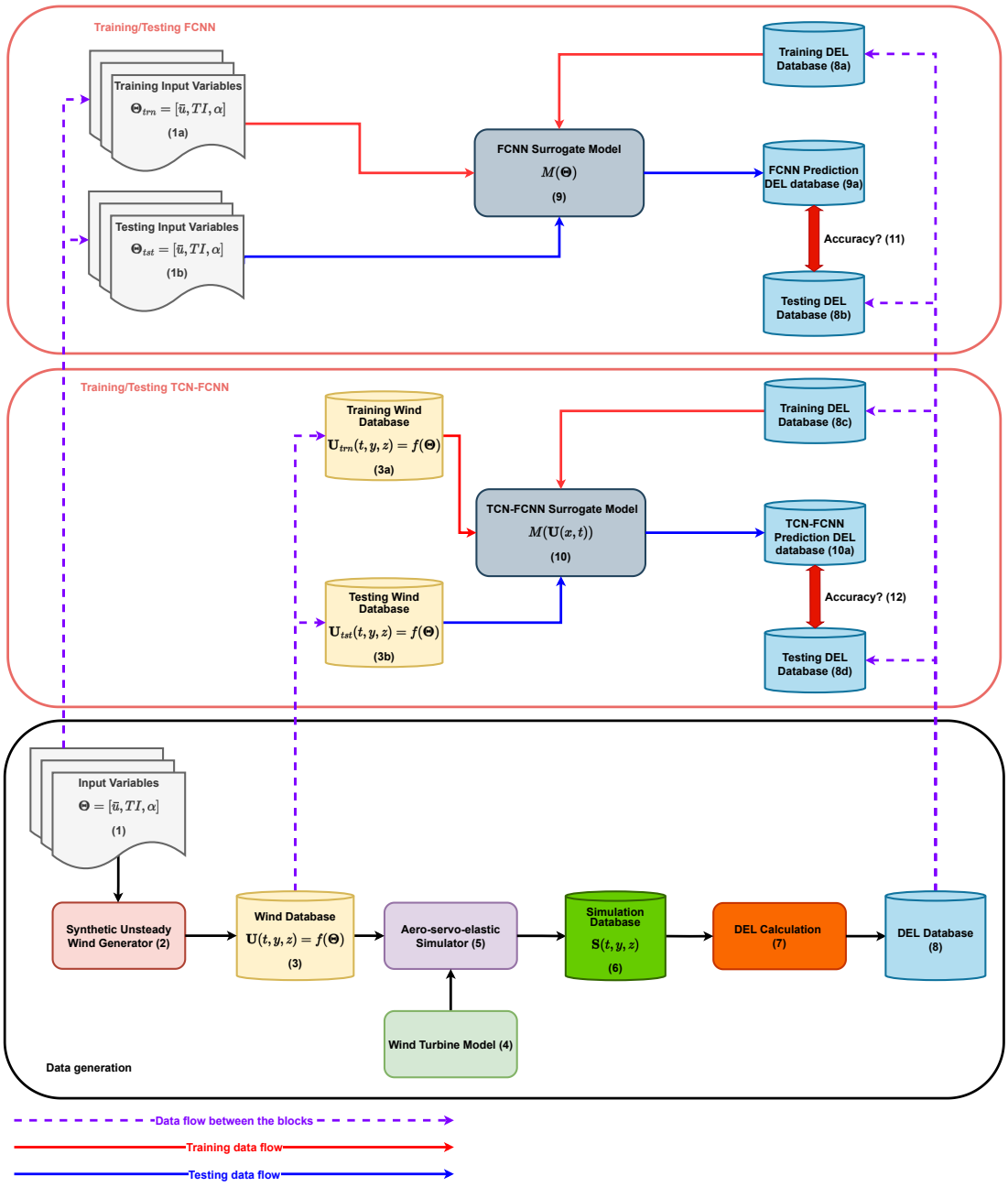


Figure 3.1: The data generation and SMs training and testing methodology

After building and training the TCN-FCNN, we will show its versatility by examining the SMs with a synthetic wind including wake time series input. In other words, we test the SMs for a turbine in the downstream wake of another turbine. We developed smaller synthetic wind time series databases with wake effects, simulation outputs and their DEL. Also, we use TL to improve the SMs' performance over the wake.

3.2.1 Fully Connected Neural Network Surrogate Model

After preparing the DEL database, we can begin training the SMs. The primary objective for the SMs is to map the input space to the output. Various mapping and regression methods are available for this task, but we suggest utilizing data-driven ML methods due to their ease of use and versatility. We developed two SM architectures; a FCNN and a TCN-FCNN. Here, the FCNN is a simple three-layer feed-forward ANN. The feed-forward ANNs are well studied and explained in the literature. For further explanation, we recommend referring to [43].

In order to train the FCNN, the input variable samples database is randomly divided into two parts: a training set comprising 90% of the samples and a testing set comprising 10% of the samples. These samples are uniquely indexed, and the training and testing sets indices are stored and tracked. The DEL database is similarly divided into training and testing sets, using the same indices as the input samples. To prevent data leakage, we ensure that there is no overlap between the training and testing databases. Once the training and testing databases are prepared, the FCNN is trained using the input variable space samples as input and DEL as output. The trained network is then tested using the testing input variables database to generate the prediction DEL. Finally, we compare the prediction with the testing DEL to measure accuracy. By following this process, we can ensure that the FCNN is accurately trained and tested, producing reliable results. Figure 3.2 shows the implemented network architecture. The input layer receives three input variables in the FCNN, while the output layer is responsible for the DEL. The weights on each neuron are determined through the training process using the weight optimizer. After the training, the FCNN is ready to predict the output based on the unseen (testing) data. Table 3.1 presents the FCNN model details.

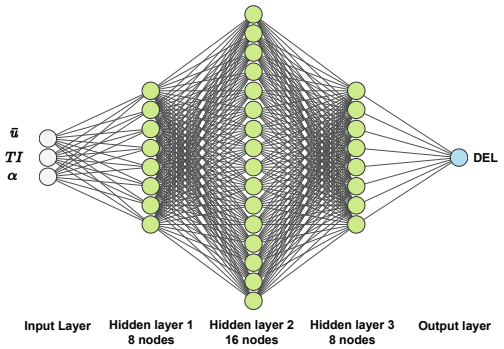


Figure 3.2: Architecture of the FCNN with three hidden layers. The number of nodes represents the implemented architecture

Table 3.1: FCNN architecture details

Property	Value
No. of hidden layers	3
No. of nodes per layer	8, 16, 8
No. of trainable parameters	321
Activation function	ReLu
Learning rate	0.001
Cost function	MSE
Training optimizer	Adam

3.2.2 Temporal Convolutional Network-Fully Connected Neural Network Surrogate Model

In this section, we explain the TCN-FCNN architecture that we used to build a SM. Firstly, we provide an overview of the key components that make up a TCN. We will demonstrate how it can be effectively combined with a FCNN.

TCN is a novel approach that utilizes the benefits of a one-dimensional convolutional neural network to perform *sequential modelling* [44]. One can define sequential modelling as a tool to map a sequential input $x_0, x_1, x_2, \ldots x_n$ to a sequential output $y_0, y_1, y_2, \ldots y_n$ as shown in Equation 3.1.

$$\hat{y_0}, \ldots, \hat{y_n} = f(x_0, \ldots, \hat{x_n}) \tag{3.1}$$

TCN is a member of the Convolutional Neural Network (CNN) family. CNNs have been used and are well known for classification proposes [45]. CNNs basics are well studied in the literature, and the interested reader is referred to [43, 45]. Research has shown that TCN is better than Recurrent Neural Network (RNN) and LSTM in terms of performance, implementation, flexibility and versatility [46, 44]. TCN is based on three main concepts: a) the length of the output and input is the same, b) data should not leak from past to future. In other words, the value of each data sequence in the output only depends on the past data sequences in the input, and c) it needs to be applicable to a long data sequence. To tackle these three, one can use

the following techniques [44]:

(a) The TCN employs a one-dimensional CNN architecture, wherein each hidden layer is of the same length as the input layer. To ensure consistent length, zero padding is incorporated in successive layers.

(b) In order to avoid data leakage, TCN utilizes *causal convolutions* architecture. In causal convolutions, the sequence n of the output solely relies on the sequences proceeding sequence n in the prior layer.

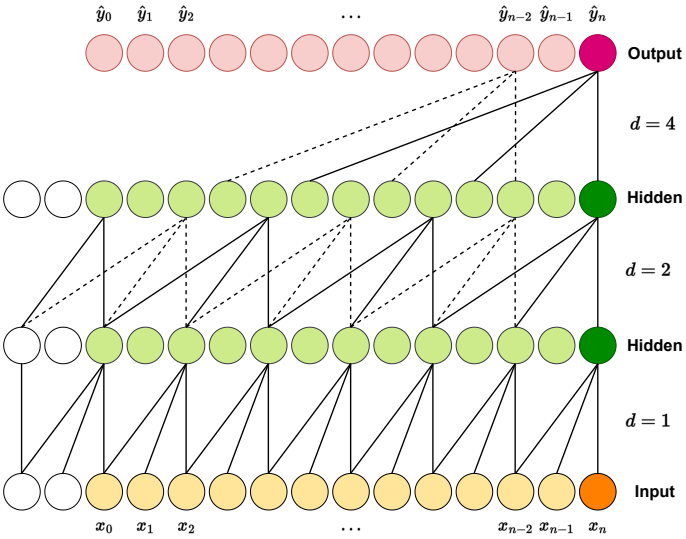
(c) For the simple causal convolutions, the length of the sequences that it can capture is a multiplication of the network depth. It makes the model deep and computationally demanding for long sequential data with vanishing gradients. The solution to this challenge is to utilize the *dilated convolution*. By using dilated convolution, the network is able to increase its receptive field significantly in an exponential manner. For a one-dimensional sequential input $\mathbf{x}$, a filter f, and the element s of the sequence, one can define the operation F as:

$$F(s) = (\mathbf{x} *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot \mathbf{x}_{s-d \cdot i} \tag{3.2}$$

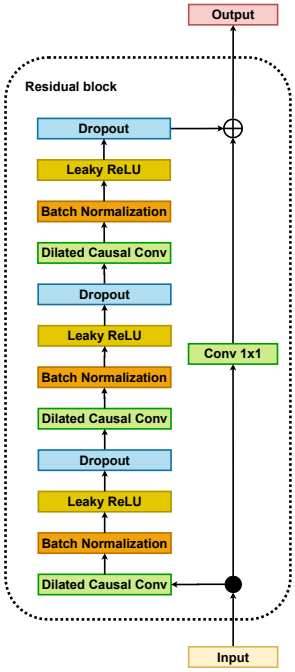
where $*$ is the convolution operator, d is the dilation factor, k is the kernel size and $s - d \cdot i$ points out the direction of the past. In dilated convolution, the dilation factor increases exponentially with the level of the network depth. Figure 3.3a shows an illustration of a dilated convolution. The history of the sequences that a layer can take into account is $(k-1)d$.

As the TCN network needs to take into account larger sequential data, it needs many layers and, as a result, gets deep quickly. This causes the network's problem of *performance degradation*, which needs to be stabilized. Therefore, we utilize a *residual block* as a replacement for a *convolutional layer* [44, 47]. The residual block methodology incorporates a branching mechanism where the input is injected into the output, passing through a CNN. The residual block used in this study is shown in Figure 3.3b.

For this study, we utilized the aforementioned TCN to extract features from the input time series. *Feature learning* or *feature extraction* is the process by which the machine learning model converts the raw data into an "internal representation", feature vector or latent space [48]. Then this feature vector is employed to detect the



(a) Illustration for a dilated causal convolution example, with kernel size $k = 3$ and dilation factors $d = 1, 2, 4$. The receptive field has the ability to encompass all values present within the input sequence. The white circles show the zero padding in the layers.



(b) TCN residual block used in this work

Figure 3.3: Dilated Causal CNN and the Residual Block for TCN

output pattern through a secondary machine learning subsystem. In this study, we took advantage of TCN ability to extract features in the sequential data. Thereafter, we used the features as the input to a FCNN. The integration of TCN and FCNN enabled us to map the wind time series into DEL. Westermann et al. used a similar approach but for a different application in [49]. The explained technique is illustrated in Figure 3.4. The "Residual Block" in Figure 3.4 made up of the components depicted in Figure 3.3b.

With all the requisite SM components in place, we can proceed with the training and testing phases. As mentioned before, we indexed the samples and TurbSim generated wind time series outputs. In the same manner as explained in Section 3.2.1, the synthetic wind time series database is divided into training and testing databases, where 90% of the database is randomly selected for the training, and the remaining 10% goes for testing. As the indices for the training and testing databases are known, they are used to divide the DEL database into training and testing databases. As the selection of the training and testing indices is random, the DEL training and testing databases members are different from the ones explained in section 3.2.1. With the

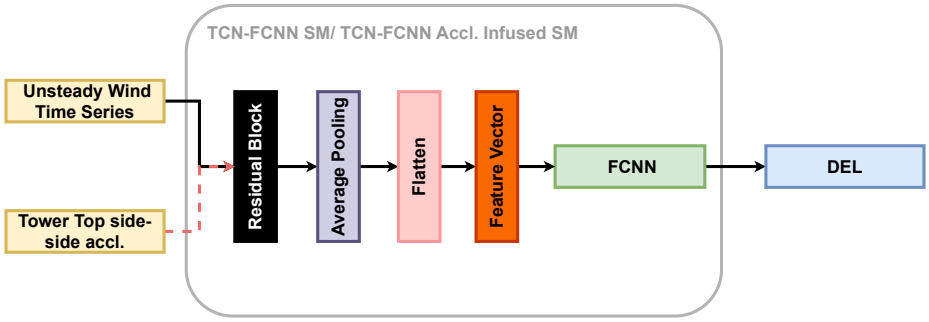


Figure 3.4: The TCN-FCNN architecture. The tower top side-side acceleration is an optional input that we will discuss further in Section 3.3.5

Table 3.2: TCN-FCNN architecture properties and details for both the main approach and TL approach. TL FCNN learning rates are for the initial training and fine-tuning consequently.

Property	Residual block	FCNN	TL FCNN
No of Conv1D filters	6, 6, 6	-	-
Kernel size	20, 13, 8	-	-
Dilation factor	1, 2, 4	-	-
Drop out rate	0.05	-	-
Average pooling size	100	-	-
Activation function	Leaky ReLU	Linear	Leaky ReLU
Trainable parameters	1950	1681	500
Number of hidden layers	-	3	3
Number of nodes per layer	-	16, 32, 16	8,11,8
Learning rate	0.001		0.001, 0.00001
Cost function	MSE		MSE
Training optimizer	Adam		Adam

training and testing databases ready, the TCN-FCNN is trained on the training data. Afterwards, we utilized the trained model to forecast DEL using the synthetic wind time series that was not included in the training database. The predicted DEL is then compared with the testing DEL to measure the accuracy of the mapping. The specifications of the TCN-FCNN SM employed have been detailed in Table 3.2.

3.2.3 Variable input space boundaries, distributions and sampling

For the data generation, selecting the appropriate input variable space, the boundaries for each variable, and their distributions is crucial. Depending on the problem at hand, different input variables might be needed. As for this study we only considered one onshore wind turbine, only the input variables that affect the wind generation are considered. These three variables are mean wind speed $\bar{u}$, turbulence intensity TI and wind shear α. Therefore, the input space $\boldsymbol{\Theta}$ can be defined as:

$$\boldsymbol{\Theta} = [\bar{u}, TI, \alpha] \tag{3.3}$$

The boundaries and distributions of our input variables help define the conditions for which our models are designed. It is important to note that wind speed is considered an independent variable, while the other two variables' boundaries and distributions depend on the wind speed. We have selected the variables, their boundaries and distributions to build the database based on research presented in [19]. Our use of QMC Sobol's sampling method [50] allows for accurate sampling of the predefined joint distributions in a deterministic non-repetitive manner. In this study, Sobol's sampling method is preferred as it is consistent and computationally efficient [51]. Also, this sampling method is reproducible and provides better uniformity properties of the samples over the distributions [52]. In the following, when we refer to the sample, it means a vector of three elements of mean wind speed, turbulence intensity and wind shear.

3.2.4 Simulation and Damage Equivalent Load databases creation

The samples from the input variable space are the input to the synthetic wind generator. Each sample from the space provides one input to the generator. The output of the generator is a "full field" synthetic wind time series [53]. The synthetic wind generation basics are explained in length in [30]. In this study, we employed TurbSim to generate the synthetic wind fields [53]. From each sample, the three input variables $[\bar{u}, TI, \alpha]$ are directly taken into the TurbSim input file and generate one synthetic wind time series using TurbSim. In order to guarantee that every time series created is distinct, a unique *seed* number is assigned to each sample. The output of the wind

generator can be defined as a function of the sample:

$$\mathbf{U}(t, y, z) = f(\Theta) \tag{3.4}$$

The output has *spatial* and *temporal* components $\mathbf{U}(t, y, z)$. The spatial component of the full field synthetic wind comes from the *grid points*, which are defined over the wind turbine rotor plane. The output of Turbsim provides one time series of synthetic wind at each grid point in x, y and z directions, namely u, v and w. These time series are correlated to each other, depending on mean wind speed and their distance from each other [30, 53].

The full-field synthetic wind is the input to the aero-servo-elastic simulations. To run these simulations, next to the synthetic wind time series, the aerodynamic model, aeroelastic model and controller model are required too. This study used an onshore model of the NREL 5MW reference wind turbine [54]. The wind turbine model includes aerodynamic, aeroelastic and controller submodules. To run the simulations, we used OpenFAST, the time marching aero-servo-elastic solver developed by NREL [55]. OpenFAST's output includes both temporal and spatial dimensions, with loads provided from various wind turbine components located at different positions, such as blades, towers, and gearboxes. This spatial aspect is integral to understanding the full scope of the data. The simulations in this study follow the IEC standards for power production Design Load Case (DLC) 1.2, as stated in IEC standards [1].

Thus far, we have established a database that comprehensively incorporates all the simulation output time series data. Once we have that, the data is processed to obtain the simulation time length statistics and DEL for evaluating the loads and fatigue. DEL calculation is based on the Palmgren–Miner linear damage rule as explained well in [56] and [57]. DEL can be formulated as:

$$DEL = \left(\frac{\Sigma n_i R_i^m}{n_{eq}} \right)^{1/m} \tag{3.5}$$

In the given context, m represents the Wöhler slope, while R_i and n_i correspond to load ranges and the respective number of cycles. The output is obtained through rainflow counting of the load time series [56]. n_{eq} is the equivalent number of load cycles which is usually the length of the simulation in s. The DEL database includes

all the calculated DELs from every simulation at its outputs.

3.2.5 Simplified Wake Model

Once the SMs are built and trained, we assess their versatility by testing them with a turbine in the wake. To proceed, we must create a new database that includes the synthetic wind and DEL with consideration given to wake effects.

The wake caused by a wind turbine has been studied extensively and is out of this manuscript's scope. Different methods and models exist to implement wakes in the aerodynamic simulation of a wind turbine [58, 59]. For the sake of simplicity and ease of implementation, we limit the study to a simplified wake definition, with the study turbine in the wake of one turbine only. The simplified wake includes a non-uniform wind speed deficit and an increase in turbulence intensity across the rotor. For implementing wakes in the synthetic wind time series, we used the method explained in [60].

For the velocity deficit caused by the wake over the rotor plane, we utilized the super-Gaussian deficit [61]. We used the formulation developed by [62] for the added turbulence intensity model. Also, the same as Bastankhah and Porté-Agel and Ishihara and Qian, we are assuming the linear expansion of the wake that occurs downstream of a turbine. The following are the steps we took to implement the downstream simplified wake model:

(1) Using Sobol's sampling method, take 2^n samples from the input variables $[\bar{u}, TI, \alpha]$ as explained in Section 3.2.3.

(2) Knowing the turbine thrust coefficient C_t at each wind speed, the ambient turbulence of the free stream and the distance between the turbines, one can calculate the downstream wake width based on the formulation in [62].

(3) With wake width calculated in the previous step, one can calculate the velocity deficit [60, 61] and added turbulence intensity [62].

(4) Both calculated velocity deficit and added turbulence intensity have spatial distribution over the rotor plane. We considered this distribution by modifying the mean wind speed and turbulence intensity of the samples for the first step. For the added turbulence intensity, the average over the rotor disk is added to

the TI value of the sample. For the deficit wind velocity, *harmonic mean* over the rotor disk is deducted from the samples mean wind speed $\bar{u}$ [60].

(5) With the *modified Sobol's samples* in our possession, we used TurbSim to generate synthetic wind time series from each modified sample. Remember that these synthetic wind time series have modified turbulence intensity and reduced mean wind speed, but the Gaussian deficit has not yet been included.

(6) For the generated synthetic wind by TurbSim, for the mean wind speed of $\bar{u}$, the wind speed in time t at each (y, z) point can be defined as:

$$U(t, y, z) = \bar{u}(y, z) + \tilde{u}(t, y, z) \tag{3.6}$$

where $\tilde{u}(t, y, z)$ is the zero mean turbulence. The inclusion of the velocity deficit caused by the wake in the generated synthetic wind can be expressed as:

$$U^*(t, y, z) = \phi(y, z) + \tilde{u}(t, y, z) \tag{3.7}$$

where $U^*(t, y, z)$ is the modified wind field and $\phi(y, z)$ is the velocity deficit distribution over the rotor yz plane, which we calculated in 3.

We will use synthetic wind with a velocity deficit and added turbulence intensity to run OpenFast simulations containing the simplified wake model and calculate the DEL as described in Section 3.2.4.

3.2.6 Transfer Learning

According to Goodfellow et al., transfer learning aims to utilize what has been learned in one context to improve the "*generalization*" in another context [43]. For this study, we use TL for the cases with wake to improve their prediction.

After following the steps outlined at the beginning of this section, we obtain trained SMs that are able to accurately predict the DEL of wind turbine components under freestream synthetic wind conditions. We then use these models to predict DEL in the wake of a turbine. We implement TL to enhance our predictions by loading the trained SMs and freezing their weights, making them untrainable. Then, we remove the FCNN part of the TCN-FCNN and replace it with a trainable FCNN.

The new FCNN we are using now has a simpler architecture compared to the one we previously used for training and testing on the free stream data. Essentially, we now have a frozen weight (untrainable) TCN along with a trainable FCNN. As with the previous training process, we utilize 90% of the wake databases for training and 10% for testing. The training has two steps; the first step is to train the aforementioned combination of untrainable TCN and trainable FCNN to the desired accuracy, and then *fine-tune* the TL model by unfreezing the TCN part weights and training it on the same data again but this time with a smaller learning rate. The properties and details of the TL FCNN are shown in Table 3.2. Figure 3.5 illustrates the architecture used for the TL.

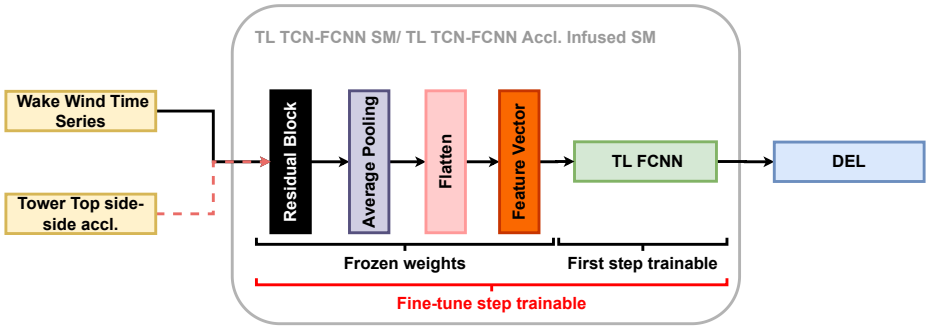


Figure 3.5: The frozen TCN with trainable simple FCNN for the first step of TL, the the fine-tuning step.

3.3 Results and discussion

To this point, we explained the background methods that have been used in this piece of work. This section will discuss the conditions that were utilized to generate the results.

3.3.1 Input variables boundaries, distributions and sampling

For generating the synthetic wind time series and DEL, we used the input variables space as it is explained in Section 3.2.3. The mean wind speed is sampled from a uniform distribution, where the boundaries are decided based on the NREL 5MW wind turbine characteristics between the cut-in to cut-out wind speed [54]. For every wind speed sample, we took a sample from the TI and α too. The other two input variables are also defined as a uniform distribution, whose boundaries are a wind

Table 3.3: The input variables boundaries

Input variable	Lower boundary	Upper boundary
Mean wind speed $\bar{u}$	$\bar{u} \geq 3m/s$	$\bar{u} \leq 25m/s$
Turbulence Intensity TI	$TI \geq 0.04$	$TI \leq I_{ref}(0.75 + 5.6/\bar{u})$
Wind shear α	$\alpha \geq \alpha_{ref,LB} - 0.23\left(\frac{\bar{u}_{max}}{\bar{u}}\right)\left(1 - \left(0.4 \log \frac{R}{z}\right)^2\right)$	$\alpha \leq \alpha_{ref,UB} + 0.4\left(\frac{R}{z}\right)\left(\frac{\bar{u}_{max}}{\bar{u}}\right)$

where
- from IEA class 1A, $I_{ref} = 18\%$
- $\alpha_{ref,LB} = 0.15$ and $\alpha_{ref,UB} = 0.22$ are reference wind shear a 15 m/s
- $\bar{u}_{max} = 25m/s$ is the upper bound of the wind speed
- R is the rotor radius, and z is the hub height

speed function. For the TI, the boundaries are based on the IEC class 1A values [1]. We choose the same boundaries for wind shear α as [19]. The input variables and their boundaries can be found in Table 3.3.

Once we have established the joint distributions and boundaries, we can generate sample points for the input variable space. As described in Section 3.2.3, we used Sobol's sampling method for this study. The Sobol's samples need to be in the order of 2^n, otherwise they lose their balance properties [64]. Therefore, we took $n = 2^{15}$ samples from the predefined distributions. We decided to have a conservative number of samples, as Sobol's sampling method enables us to reduce the number of samples without losing the benefits of the method or resampling the domains. To generate an example of the variable space, we took $2^{10} = 1024$ samples from the predefined distribution in Table 3.3. The samples and the input variable boundaries are displayed in Figure 3.6.

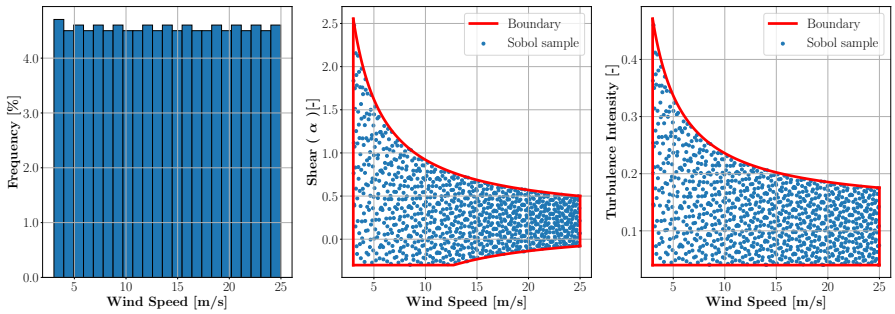
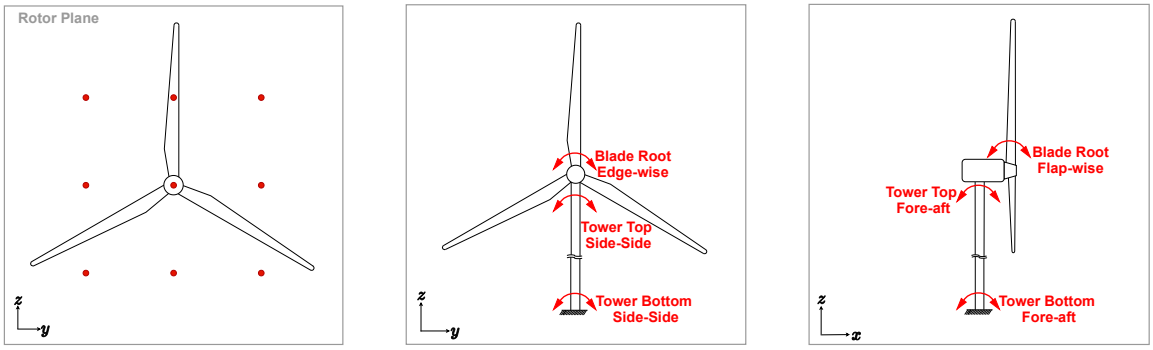


Figure 3.6: 1024 Sobol's samples from the predefined distributions for $\bar{u}$, TI and α with the boundaries of the variables.

3.3.2 Turbsim and OpenFAST output

As mentioned before, we use each sample from the input variables to generate a unique synthetic wind time series using TurbSim. Every TurbSim output generated from each sample is assigned a unique seed number to ensure that there are no repeating seeds. The TurbSim output format is well described in [53]. For this study, we used a 15 by 15 grid over the rotor's plane. For training and testing purposes, we only took into account nine synthetic wind time series in x direction out of 225 synthetic wind time series. Our tests show including the wind in y and z directions would not improve the training or testing results; therefore, they are omitted. These nine synthetic wind time series are approximately located at the middle of the rotor and hub height. Our tests show other configurations of the points (e.g. circular layout of points) have little impact on the results. The grid points are selected to be roughly located at the blades' mid-span. This selection is illustrated in Figure 3.7a. Regarding the time component, the synthetic wind time series has a frequency of $20Hz$, with a duration of $720sec$. After running the simulation and later in the training/testing step, we upsample the synthetic wind to $1Hz$ due to the hardware constraint.



(a) TurbSim output

(b) OpenFAST output channels y, z plane

(c) OpenFAST output channels x, z plane

Figure 3.7: (a) Illustration for selected TurbSim output grid point locations as the input to the SM for training and testing. (b) and (c) show a schematic drawing of a turbine, with the output load channels.

As mentioned before, we run aeroelastic simulations on an onshore NREL 5MW model using OpenFAST. OpenFAST can provide an extensive set of outputs, namely channels, at different components of the turbine. The channels and their descriptions

can be found in [2]. For this study, we took into account six moment output channels and the average generated power for the training/testing objectives. These six moments are blade root edgewise and flap-wise moments, tower top fore-aft and side-side moments, and tower bottom fore-aft and side-side moments. Figure 3.7b and 3.7c illustrate a schematic drawing of the wind turbine with the load channel locations that we used for training/testing in this study. The OpenFAST output channel label and its corresponding naming for this study are provided in Table 3.4.

Table 3.4: Channel label, naming and units

OpenFAST Channel label	Naming	Post Processing	Unit
GenPwr	Generated Power	10 minutes average	[kW]
RootMxb1	Blade edgewise moment	DEL	[kNm]
RootMyb1	Blade flapwise moment	DEL	[kNm]
YawBrMxp	Tower-top side-side	DEL	[kNm]
YawBrMyp	Tower-top fore-aft	DEL	[kNm]
TwrBsMxt	Tower bottom side-side	DEL	[kNm]
TwrBsMyt	Tower bottom fore-aft	DEL	[kNm]

We run $n = 2^{15} = 32768$ aeroelastic simulations using OpenFAST for this study. We run the simulations in 2048 batches of 16 simulations in parallel using Digital Research Alliance of Canada resources. Each simulation ran for $720s$, but the first $120s$ of the simulation output was discarded to avoid any initialization effect. The time step for the aeroelastic simulation was set to $0.00625s$, while the output resolution is $20Hz$. After running all the simulations and building the simulations output database, we calculate the DEL for each simulation, for the interested output channels for Wöhler slope $m = 4$ and $n_{eq} = 600$ in Equation (3.5). Also, we considered the 10-minute average of the generated power. To read the OpenFAST output files and calculate the DEL, we used python `pyfast` library [65].

3.3.3 Turbine in wake output

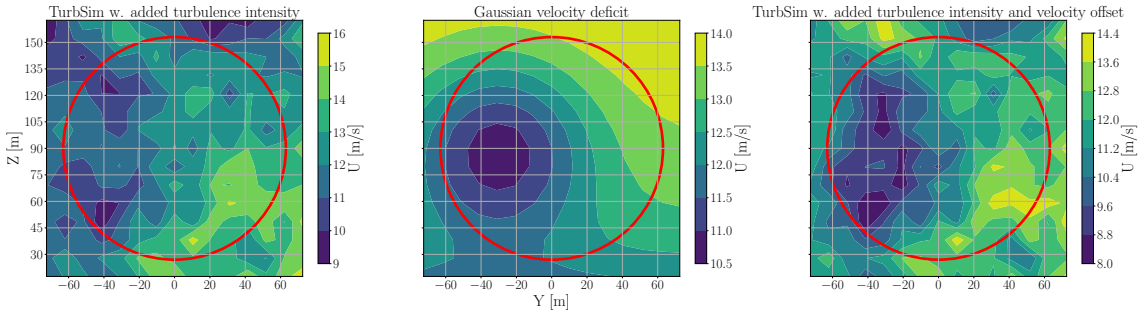
Our aim is to test the effectiveness of the trained TCN-FCNN SMs by using wake input. Specifically, to determine if the model, which is trained on a turbine in the free stream, can accurately predict the DEL of a turbine in wake as well. The test scenario involves one turbine in free stream, which the SMs are trained on, and one turbine in downstream wake. The distance between the two turbines is $7D$, where

D represents the rotor diameter. In our specific case, we are considering the rotor diameter of $126m$ for NREL 5MW, which results in a distance of $882m$ between the two turbines.

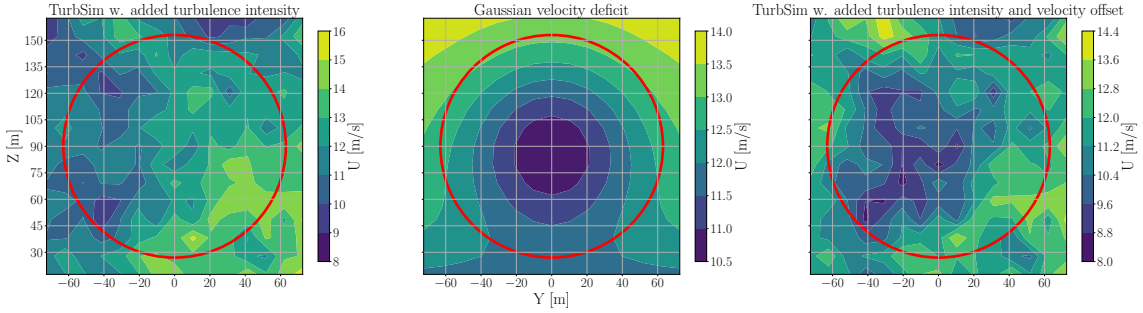
We followed the same process described in Section 3.3.1 by taking 2048 samples from the distributions outlined in Table 3.3. To calculate the wind velocity deficit and add turbulence, we used the Gaussian model [61] and [62], respectively, as explained in Section 3.2.5. We then adjusted the $\bar{u}$ and TI of each sample based on the harmonic mean of the wind velocity deficit and the arithmetic mean of the added turbulence intensity over the rotor. Using the modified samples, we generated 2048 Turbsim full-field outputs, following the process explained in Section 3.3.2. With the Turbsim output now available with modified $\bar{u}$ and TI, we utilized a Python script to offset the generated synthetic wind with the Gaussian velocity deficit profile [60].

In Section 3.2.5, it was explained that the wind velocity deficit has a distribution across the yz plane. This distribution can shift across the rotor plane depending on the location of the wake centre. We established three wake cases with wake centres located at $(-30m, 90m)$, $(0m, 90m)$, and $(30m, 90m)$ on the rotor plane yz. The wake centre is assumed not to move vertically since both turbines have the same hub height of $90m$. Figure 3.8 illustrates an example of the velocity deficit effect on the TurbSim output. In Figure 3.8, the first column shows the TurbSim output with the added turbulence intensity, the middle column is the Gaussian velocity deficit for the aforementioned wake centers, and the last column is the first column with the velocity deficit offset. They are all a snapshot of the TurbSim output at $320s$, and the input samples are $[\bar{u} = 13.42m/s, TI = 11\%, \alpha = 0.107]$. The red circle is the rotor disk.

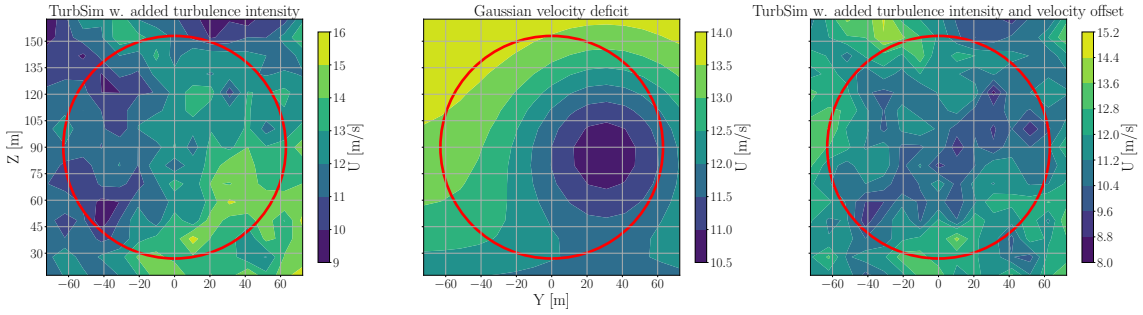
As all the 2048 TurbSim outputs are at hand, one can run OpenFast simulations and calculate DEL as explained in Section 3.3.2.



(a) Wake centre at $(-30m, 90m)$



(b) Wake centre at $(0m, 90m)$



(c) Wake centre at $(30m, 90m)$

Figure 3.8: The velocity deficit implementation on a TurbSim output with the added turbulence intensity.

3.3.4 Training-Testing

Now that we have constructed all the necessary databases, we can begin the training process. We must first normalize the data before training the SMs. In this study, we employ *min-max scaling* for both input and output values, scaling them to a range of 0 to 1. The input variables are scaled individually, while all the synthetic wind time series, regardless of the mean wind speed, are included in a single scaling procedure. During the training process of the TCN-FCNN and FCNN models, we implemented a separate scaling of the DEL of each output channel. This approach was necessary to ensure the scaling was tailored to the specific needs of each channel. A total of twelve SMs were trained, with six models being trained for each respective type.

The total number of samples in the dataset is 32768. As mentioned in Section 3.2.2, this dataset is split into two parts for different purposes randomly by choosing the indices of the samples through a non-repetitive random number generator. The training set contains 90% of the samples, and the testing set contains 10% of the samples. This means that the training dataset includes 29492 full $600s$ wind time series and its corresponding DEL on wind turbine components, and the remaining 3277 are in the training dataset. Rather than training the SMs on all the training data, the training data set is divided into batches of 256 samples. Afterwards, the SMs trained on each batch, and after going through all the batches of the training, one "epoch" is completed. The main optimization method used in ML community is Stochastic Gradient Descent (SGD) [66]. Taking all the training data set for the training while employing SGD requires a large amount of memory, and SGD may land you at a "saddle point" [67]. One can tackle both of these issues in training by dividing the training data into batches, as explained before. One disadvantage of this method is that it requires more epochs for the model weights to be fully trained and converged. In this study, we used Python package `tensorflow` for the ML model development, training, and testing [68]. The table outlining the settings and details for training in `tensorflow` can be found in Table 3.5.

We employed early stopping for the training as it reduces the required training time. Once the training phase is complete, the remaining 10% of the data that was not used during training is utilized for testing purposes. The output of the testing procedure provides the accuracy of the fitted models. For this study, we use *coefficient*

Table 3.5: Training setting for the SMs

Parameter	TCN-FCNN	FCNN	TL TCN-FCNN
Batch size	256	256	64
Trainable parameters	3631	321	500
Maximum number of epochs	3000	3000	3000
Validation split	5%	5%	5%
Early stopping - monitoring	validation loss	validation loss	validation loss
Early stopping - patience epochs	300	300	3000
Early stopping - best weight restoration	True	True	True
Input shape	(batch size, 600, 9)	(batch size, 3)	(batch size, 600, 9)

of detemination, R^2, and NRMSE as the measures for the fitted models' accuracy.

3.3.5 TCN-FCNN results

This section provides the output of the testing process, as explained before, for the TCN-FCNN SMs. Figure 3.9 shows the results of the testing on the trained SMs for each output channel scaled DEL. Each plot in the figure delivers one channel, and data connects to the mean wind speed of the samples using a colour map. Upon reviewing the outcomes, it is evident that the SMs offer precise prediction based on synthetic wind time series data that it has not previously encountered. Based on the colour maps, it is inconclusive to determine the correlation between the input variables of the sample and the accuracy of the fit.

Although all channels have high R^2 values, the values decrease as we move from the blade root moments downwards. This decrease can be explained by the physical problem we are dealing with. The SMs have only one input, and they map wind input time series in x direction to DEL outputs. The loads closer to the rotor are more affected by the wind input, while the structural dynamics of the wind turbine influence the further loads. Both fore-aft and side-side moments exhibit similar behaviour, but fore-aft moments are predicted more accurately. Therefore, it is reasonable to assert that the fore-aft moments are predominantly attributable to the wind, whereas the structural dynamics more significantly influence the side-to-side moments. To test this hypothesis, we infused the input of the TCN-FCNN input with the tower top side-side time series acceleration a_{TTy}. In other words, the TCN-FCNN maps the combination of synthetic wind time series *and* tower top side-side acceleration time series to DEL. Figure 3.4 visually represents this network with tower top side-side acceleration as an optional input.

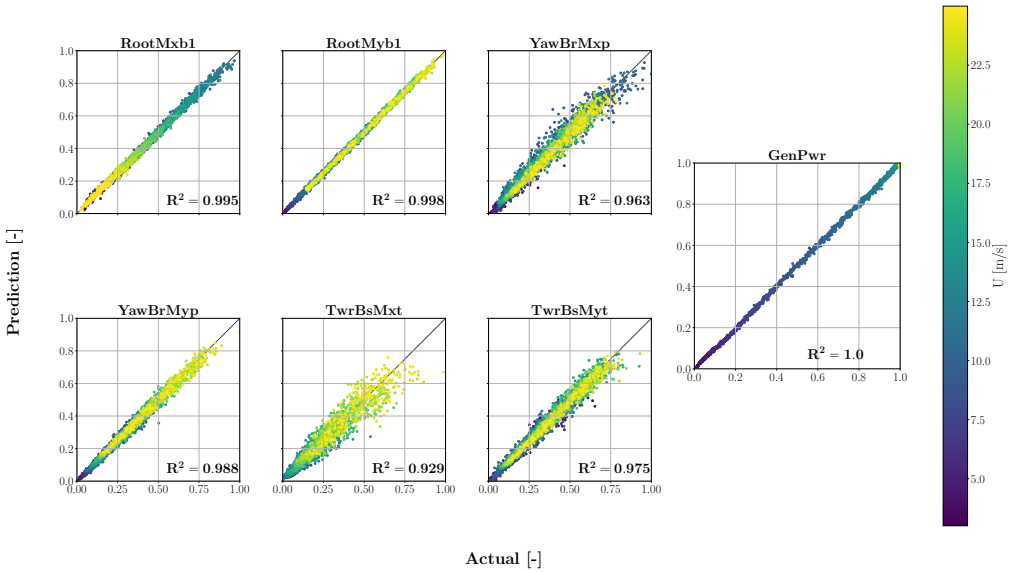


Figure 3.9: Testing results for TCN-FCNN trained SMs. The colour map represents a range of changes in the mean wind speed of the samples.

The modified SMs used for this are identical to those shown in Tables 3.2 and 3.5. The only difference is that the third dimension of the input shape in Table 3.5 has changed to 10 due to the concatenated time series. We followed the same process for training and testing these acceleration-enhanced SMs as we did for the original ones. In Figure 3.10, the data indicates that incorporating the tower top acceleration time series into the input led to a better fit for the side-side moments, particularly for the tower bottom side-side moment. This confirms our hypothesis that these SMs can accurately capture the physics of the model at hand. One may argue that including the wind time series in the y direction in the input would improve the tower bottom side-side moment R^2 value. We tested this hypothesis, but it did not improve the accuracy of the TCN-FCNN model.

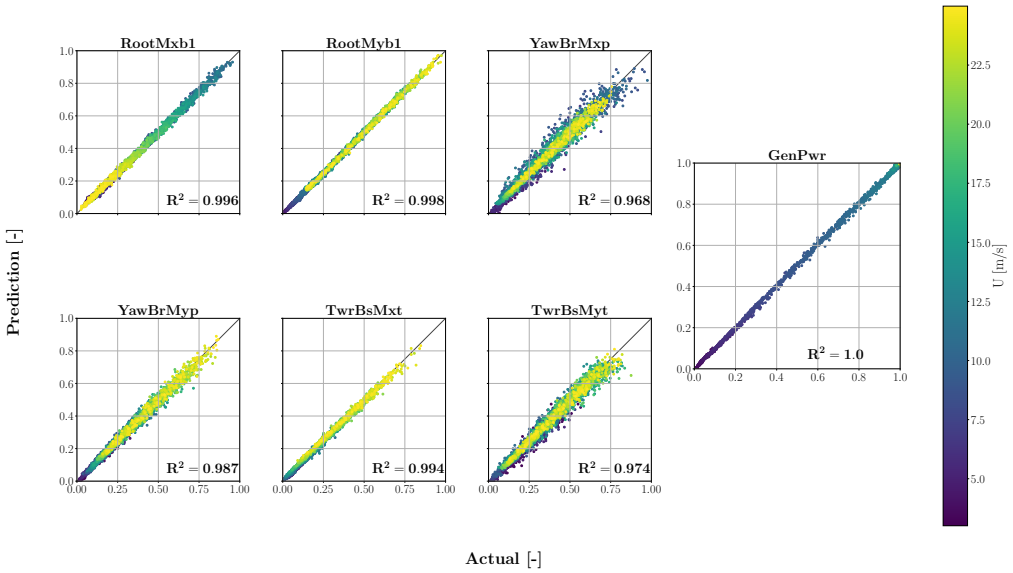


Figure 3.10: Testing results for TCN-FCNN trained SMs. The input for these SMs is infused with the tower top acceleration time series. The colour map represents a range of changes in the mean wind speed of the samples

3.3.6 FCNN results

As mentioned in Section 3.3.4, we trained and tested the FCNN SM on the three input variables samples, namely wind speed, wind shear and turbulence intensity. The FCNN aims to map the three input variables to DEL. This is very similar to the approach that was employed in [40]. The results are presented in Figure 3.11. Similar to the TCN-FCNN results, the R^2 values decrease from the top to the bottom of the turbine. Considering the simplicity of the FCNN SMs, they perform very well in the testing phase.

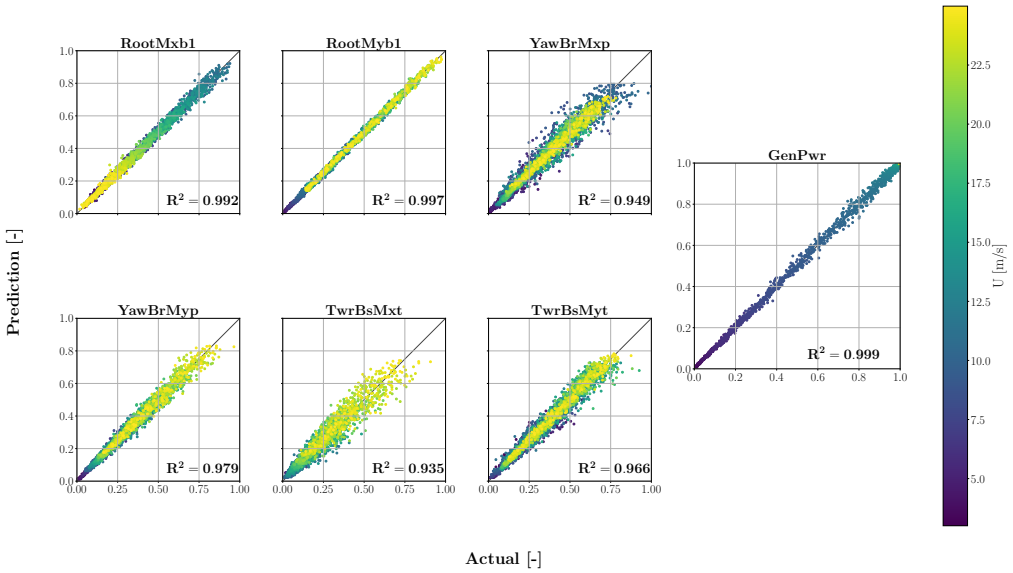


Figure 3.11: These are the testing results for the FCNN on the trained SMs.

3.3.7 Results comparison

To this point, we trained three different types of SMs on our data set. These SMs showed a good ability to predict the DEL from a limited amount of sequential data or input variable data. Each SM architecture has its own advantages and disadvantages. For TCN-FCNN architecture, the SMs can digest the complexities of wind time series. However, the model is complex and loses accuracy for the channels that are not close to the rotor. On the other hand, FCNN architecture is simple and cheap to train, while it has higher R^2 values for the channels below the rotor. However, this model suffers from the same challenge in terms of R^2 values decline, and it is not prepared to take time series and needs input variables that may not be available all the time. To make the comparison more straightforward, the R^2 value for power and output channels and the required Graphics Processing Unit (GPU) time for the training is provided in Table 3.6.

The data presented in Table 3.6 reveals that all three SMs have consistently produced high R^2 values and low NRMSE. Notably, the utilization of tower top acceleration as the added input to wind has significantly enhanced the accuracy of prediction for the tower bottom side-side moment. This improvement is a testament to the TCN-FCNN SMs' ability to comprehend the mechanics behind the input-output cor-

Table 3.6: The accuracy of the fit and computational time for the SMs training. NRMSE is in percentage.

	TCN-FCNN			TCN-FCNN Accl. infused			FCNN		
Channel	R^2	NRMSE	GPU Time [Sec]	R^2	NRMSE	GPU Time [Sec]	R^2	NRMSE	GPU Time [Sec]
GenPwr	1.000	0.006	442	1.000	0.005	337	0.999	0.010	271
RootMxb1	0.995	0.018	578	0.996	0.016	594	0.990	0.031	379
RootMyb1	0.998	0.009	805	0.998	0.010	689	0.998	0.013	397
YawBrMxp	0.968	0.140	800	0.972	0.126	1042	0.987	0.190	110
YawBrMyp	0.987	0.039	807	0.986	0.040	781	0.996	0.064	397
TwrBsMxt	0.936	0.153	363	0.994	0.013	274	0.944	0.156	106
TwrBsMyt	0.971	0.094	613	0.977	0.099	756	0.977	0.119	163

relation. Regarding the computational cost, FCNN SMs are more efficient for all the outputs.

When examining Table 3.6, one might question the purpose of developing the TCN-FCNN SMs as they are more complex and computationally expensive. The TCN-FCNN approach offers a significant benefit by examining the wind's time series rather than solely its statistical properties. The DEL results from wind and/or wave time series oscillations. If we were to reduce these oscillations solely to wind or wave input statistics, this would undermine the accuracy of the DEL prediction. However, the TCN-FCNN can incorporate these oscillations and map them to the DEL. One challenge here is to free the input from the time series' length, which is not within the scope of this study. We will explore this further in our future studies. We briefly discuss the simulation length effect in Section 3.3.10.

We underline that the TCN-FCNN model can effectively decompose the wind field into its constituent features, which include the input variables. This capability was tested by expanding the feature vector in Figure 3.4 with the three input variables. Even with this augmentation, the R^2 values remained consistent, reaffirming the robustness of the TCN-FCNN in characterizing the wind field. Essentially, a latent space has been identified by the TCN suitable for accurate DEL prediction by the FCNN stage.

We tested the TCN-FCNN architecture to assess its ability to handle ultimate loads. During our analysis, we found that the SMs could accurately predict ultimate loads with a comparable level of precision as DEL prediction.

3.3.8 TCN-FCNN SMs in wake with TL

Considering the methodology employed for incorporating wake effects into our simulations, the use of a FCNN proves ineffective in this context. The FCNN relies on input parameters such as mean wind speed, turbulence intensity TI, and wind shear α, which cannot adequately capture the complexities of wake interactions, as they cannot be condensed into a single scalar value. In a study by Dimitrov, an FCNN-based Surrogate Model was utilized to model wake effects. It was noted that their model required additional inputs depending on the wind farm layout. In contrast, the TCN-FCNN approach, which relies solely on the flow information at the turbine location, demonstrates the capability to address wake challenges without necessitating additional inputs, provided that the flow characteristics over the turbine are well-defined.

After training the TCN-FCNN and FCNN SMs, we tested the SMs on the input with the wake. The initial results without any TL did not provide an accurate prediction. Therefore, we used the TL as explained in Sections 3.2.5 and 3.3.3. Table 3.5 details the training setting for the TL models. The training is done on the 90% of the turbine in wake databases, and testing is based on the remaining 10%. We used both normal TCN-FCNN and acceleration infused TCN-FCNN in the TL for the turbine in the wake. The results for three wake centers are presented in Table 3.7.

Table 3.7: The results for the turbine in wake SMs in predictions after going through two stages of TL. The TL is done on both TCN-FCNN and the acceleration infused TCN-FCNN. For the sake of space acceleration infused TCN-FCNN indicates as TCN-FCNN A.I. NRMSE is in percentage.

Wake centre	(-30m,90m)				(0m,90m)				(30m,90m)			
	TCN-FCNN		TCN-FCNN A.I.		TCN-FCNN		TCN-FCNN A.I.		TCN-FCNN		TCN-FCNN A.I.	
Channel	R^2	NRMSE	R^2	NRMSE	R^2	NRMSE	R^2	NRMSE	R^2	NRMSE	R^2	NRMSE
GenPwr	0.999	0.998	0.999	1.214	0.998	1.949	0.999	1.002	0.999	1.275	0.999	0.996
RootMxb1	0.970	3.546	0.972	3.654	0.961	4.381	0.918	5.281	0.969	3.689	0.971	3.642
RootMyb1	0.979	3.335	0.972	3.580	0.975	3.793	0.962	4.844	0.968	4.234	0.976	4.636
YawBrMxp	0.952	4.787	0.953	4.617	0.960	3.808	0.938	5.756	0.960	4.560	0.955	4.690
YawBrMyp	0.925	6.783	0.960	5.104	0.923	7.593	0.962	4.899	0.909	7.957	0.950	5.540
TwrBsMxt	0.804	10.750	0.994	1.488	0.814	10.466	0.992	1.982	0.804	10.871	0.992	1.573
TwrBsMyt	0.858	7.837	0.944	4.312	0.911	5.643	0.944	5.310	0.939	5.343	0.955	4.927

The results predicted in Table 3.7 follow a similar trend to those obtained from the free stream turbine for all three wake centers. However, the SMs infused with

acceleration provide higher R^2 values for almost all cases compared to the free stream, where the effect was mainly limited to tower bottom channels. Since the turbine's behaviour is more complex in the wake, knowledge of its structural dynamics can be more influential in prediction. Therefore, including a channel from the turbine structure can aid the SMs in training a more accurate model and providing better predictions. It is worth noting that this training is conducted on only 1844 out of 2048 data points in the wake database, which is relatively small. Despite this, the model's ability to have a low NRMSE and high R^2 values demonstrates its strength.

3.3.9 How much data is enough data?

One question that needs to be answered is how much data is enough to train these SMs accurately. In other words, we need to determine if reducing the number of sample points will affect the accuracy of the predictions made by the SMs. As we used the QMC method for sampling in this study, we can easily decrease the number of samples without having to redo the simulations. However, as our samples are based on Sobol's samples, we need to stick to the 2^n rule. To determine the amount of data needed, we trained the SMs on a smaller number of samples ranging from 2^6 to 2^{14}. Then, we randomly selected 1000 samples from the remaining input samples for prediction. For example, in the case of 2^{10}, we trained the SMs on 1024 Sobol's samples data points and randomly selected 1000 samples from the remaining $2^{15} - 2^{10}$ samples for prediction. This enabled us to ensure fairness in comparing the R^2 values without any data leakage. The sensitivity analysis results in Figure 3.12 indicate that the R^2 value remains relatively high across all channels until the number of samples is reduced to 2^8.

This shows that the SMs are versatile and do not require many sample points to make accurate predictions. The versatility of the SMs can be attributed to the simplicity and power of the models for providing accurate predictions and the effectiveness of Sobol's sampling in covering the input variable domains even with a low number of data points. This coverage helps the SMs interpolate well between the data they are not trained on. The displayed data in Figure 3.12 indicates that FCNN SMs exhibit less sensitivity to the number of samples. This observation aligns with the expectation that FCNN SMs possess a simpler architecture and fewer parameters

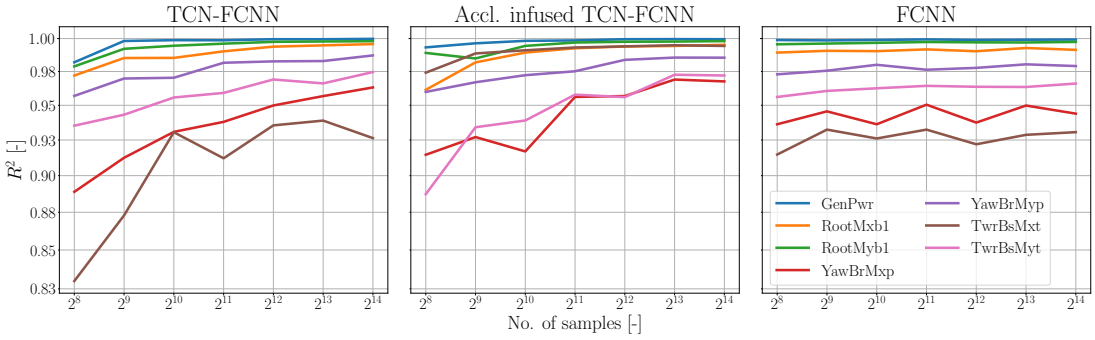


Figure 3.12: Sensitivity of R^2 to the number of training samples. The y-axis is on a logarithmic scale with a base of ten, while the x-axis is on a logarithmic scale with a base of two.

to be trained. Therefore, the trained model improvement is minimal after passing the threshold of the number of samples. In the TCN-FCNN SMs, the improvement of the R^2 value varies depending on the channel. The tower channels exhibit a greater rate of R^2 improvement as compared to the rotor channels, with the side-side moment channels of the tower being the most prominent example. Additionally, acceleration infused TCN-FCNN shows higher improvement rates for tower top side-side and tower bottom fore-aft channels.

3.3.10 Time series length and data augmentation

In this study, we used a ten-minute time series for both input and output of the OpenFAST simulation. This is a common practice in the wind turbine engineering field as recommended by standards [1]. Yet from the perspective of LiDAR and wind turbine controller, a ten-minute time series may be relatively long. Therefore, we aimed to include the ability to handle shorter synthetic wind time series and map them to DEL, which would be more attractive. Besides, in the field of machine learning, having enough training data is a challenge, and data augmentation is a solution [69]. To satisfy these two purposes, we attempted to augment data by dividing the 600-second synthetic wind and load time series into shorter segments and calculating DEL from those segments. If possible, the goal was to use even less than the minimum required number of simulations. However, our investigations have revealed that subsampling the 600-second load time series into segments shorter than 300 seconds adversely affects the accuracy of the DEL calculation. This is due to missed cycle

counting of a the shorter data length obtained from OpenFAST output. Hence, this approach was not included in this manuscript.

3.4 Conclusion

This study explores the potential of employing a sequential ML model to develop a SM that correlates high-resolution wind time series with the DEL of wind turbine components. The methodology utilized in this manuscript involves creating a TCN-FCNN architecture for mapping synthetic wind time series to DEL, alongside a simpler FCNN for comparative purposes. We divided our methodology into twelve stages, including specifying the input variable space, generating synthetic wind time series, conducting aero-servo-elastic simulations, calculating DEL, splitting the data into training and testing databases, and building SMs. Also, we build a database of synthetic wind time series and DEL for a turbine in wake to test the versatility of the TCN-FCNN SMs using TL.

Our work begins with defining the input variable space and determining their boundaries, distributions, and sampling methods. We use a QMC Sobol's sampling technique to generate non-repetitive samples, guaranteeing uniformity, traceability and reproducibility. Next, we continued with the generation of synthetic wind time series using TurbSim, based on the input variable samples. These time series are stored in the Wind Database, forming the basis for subsequent simulations. The aero-servo-elastic simulations are performed using an NREL 5MW reference wind turbine model and OpenFAST, following IEC standards for power production DLC 1.2 [1]. The simulation output is stored in the Simulation Database, providing load time series data for various wind turbine components. With the loads time series data at hand, we calculate the DEL for each wind turbine component, adhering to the Palmgren-Miner linear damage rule. To train and test the SMs, we split the DEL database into training and testing sets, while ensuring no overlap between them. Two SM architectures were developed: a simple FCNN and a more complex TCN-FCNN. Both models are trained and tested to predict DEL based on unseen input variables or synthetic wind time series data. The FCNN SM serves as a benchmark for comparison with the more advanced TCN-FCNN SM. By comparing the accuracy and performance of these models, we gain insights into the effectiveness of our approach. Moreover, we introduced the concept of testing our SMs in the context of a wake

scenario. We created a new dataset that considers the wake effect on a wind turbine by implementing simplified wake models, thus expanding the versatility of our SMs.

In the results section, three different SM architectures were investigated: TCN-FCNN, FCNN, and an enhanced version of TCN-FCNN infused with tower top acceleration time series. The TCN-FCNN architecture was designed to take advantage of the wind time series data, making it capable of capturing the complex temporal dependencies in the wind field. Yet, its performance varied for different output channels, with higher R^2 values obtained for loads closer to the rotor and decreasing accuracy for loads influenced by structural dynamics. The addition of tower top acceleration time series as an input feature improved the accuracy of predicting the tower bottom side-side moment, demonstrating the SM's ability to discern relevant physics. In contrast, the FCNN architecture, which solely relies on input variables, offered a simpler and more efficient model with competitive predictive accuracy. The FCNN SMs performed well for all output channels, with R^2 values having an inverse relationship with the distance from the rotor. We further analyzed the trained TCN-FCNN models to determine how well they can predict DEL for a turbine situated in the downstream wake, a use case the FCNN SM could not tackle and illustrative of the TCN-FCNN architecture motivation. Our findings indicate that by using TL on a limited dataset, we can accurately forecast the DEL of a turbine in a wake. A sensitivity analysis was conducted to determine the minimum required sample size for training the SMs. It was found that both TCN-FCNN and FCNN SMs remained accurate with a relatively small number of samples, making them versatile and efficient for practical applications such as wind farm layout optimization. The choice between TCN-FCNN and FCNN SMs depends on the specific application requirements. TCN-FCNN is suitable when capturing fine-grained temporal dependencies in the wind field is crucial, while FCNN offers a simpler and more computationally efficient alternative with competitive performance. These SMs provide valuable tools for predicting DEL and enhancing wind turbine reliability, reducing the need for extensive and expensive sets of simulations. One of the drawbacks of this work is the input time signal length. One needs to investigate the possibility of liberating the SM from this time constraint, as it would make the model more versatile and more applicable to the wind speed time series of any length.

3.4.1 Future work

This study is the initial phase of building a ready-to-use and generalizable SMs for wind turbines. In this manuscript, we explore the ML based SM for this purpose and specifically TCN application in wind turbine engineering, while there is an extensive scope for further investigation. In future studies, we will implement TCN-FCNN methodology on an offshore wind turbine introducing complicated wave loading magnitude and directional spectra. Also, we will investigate the possibility of extending TL in a wind farm to train the SMs on one turbine and use transfer learning to build SMs for others in different wake conditions quickly. In this study, we took nine wind time series from the synthetic wind field as the input; therefore, reducing the number of wind time series is another interesting investigation alongside optimization of the placement of the points as a hyperparameter.

Unfortunately, we were unable to access high-resolution wind turbine measurement data for our study. However, we recognize that incorporating this type of data into our methods could greatly enhance our research and should be a focus for future work. Additionally, we acknowledge that the synthetic data and mathematical models used in our SMs may not be as accurate as reality. As the saying goes, "All models are wrong, but some are useful." While the models that we use to build the databases and train the SMs may not be perfect, they still hold value. Therefore, one idea for future work is that the trained SMs can be applied effectively on high-resolution measurement data by utilizing TL and inserting them between two trainable layers at the input and output stages. This approach can prove especially helpful when faced with limited measurement data.

Code and data vailability

The code supporting this study's findings is available from the corresponding author, RH, GitHub page. The data is available upon request.

Competing interests

The authors confirm there are no competing interests present

Acknowledgements

We greatly acknowledge the funding for this study by the Natural Sciences and Engineering Research Council of Canada (NSERC). This research was partly enabled by support provided by the Digital Research Alliance of Canada (alliancecan.ca).

Bibliography

[1] IEC 61400-1. Wind energy generation systems - Part 1: Design requirements. Standard, International Electrotechnical Commission, 2019.

[2] Jason M Jonkman, Marshall L Buhl Jr, et al. FAST user's guide. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2005.

[3] Torben J Larsen and Anders Melchior Hansen. How 2 HAWC2, the user's manual. 2007.

[4] EA Bossanyi. GH bladed theory manual. *GH & Partners Ltd*, 2:56–58, 2003.

[5] Bruno Sudret. *Uncertainty Propagation and Sensitivity Analysis in Mechanical Models–Contributions to Structural Reliability and Stochastic Spectral Methods*. PhD thesis, 2007.

[6] B.A. Williams and S. Cremaschi. Surrogate Model Selection for Design Space Approximation And Surrogatebased Optimization. In *Computer Aided Chemical Engineering*, volume 47, pages 353–358. Elsevier, 2019. ISBN 978-0-12-818597-1. doi: 10.1016/B978-0-12-818597-1.50056-4.

[7] M. J. Asher, B. F. W. Croke, A. J. Jakeman, and L. J. M. Peeters. A review of surrogate models and their application to groundwater modeling. *Water Resources Research*, 51(8):5957–5973, 2015. ISSN 1944-7973. doi: 10.1002/2015WR016967.

[8] Dongbin Xiu and George Em Karniadakis. The Wiener–Askey Polynomial Chaos for Stochastic Differential Equations. *SIAM Journal on Scientific Computing*, 24(2):619–644, 2002. ISSN 1064-8275. doi: 10.1137/S1064827501387826.

[9] Thierry Crestaux, Olivier Le Maı̂tre, and Jean-Marc Martinez. Polynomial chaos expansion for sensitivity analysis. *Reliability Engineering & System Safety*, 94(7):1161–1172, 2009. ISSN 0951-8320. doi: 10.1016/j.ress.2008.10.008.

[10] A. O'Hagan. Curve Fitting and Optimal Design for Prediction. *Journal of the Royal Statistical Society: Series B (Methodological)*, 40(1):1–24, 1978. ISSN 2517-6161. doi: 10.1111/j.2517-6161.1978.tb01643.x.

[11] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass, 2006. ISBN 978-0-262-18253-9. doi: https://doi.org/10.7551/mitpress/3206.001.0001.

[12] Chong Wang, Xin Qiang, Menghui Xu, and Tao Wu. Recent Advances in Surrogate Modeling Methods for Uncertainty Quantification and Propagation. *Symmetry*, 14(6):1219, 2022. ISSN 2073-8994. doi: 10.3390/sym14061219.

[13] Jakub Kudela and Radomil Matousek. Recent advances and applications of surrogate models for finite element method computations: A review. *Soft Computing*, 26(24):13709–13733, 2022. ISSN 1433-7479. doi: 10.1007/s00500-022-07362-8.

[14] Armin Dadras Eslamlou and Shiping Huang. Artificial-Neural-Network-Based Surrogate Models for Structural Health Monitoring of Civil Structures: A Literature Review. *Buildings*, 12(12):2067, 2022. ISSN 2075-5309. doi: 10.3390/buildings12122067.

[15] Gang Sun and Shuyue Wang. A review of the artificial neural network surrogate modeling in aerodynamic design. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 233(16):5863–5872, 2019. ISSN 0954-4100. doi: 10.1177/0954410019864485.

[16] Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993. ISSN 0893-6080. doi: 10.1016/S0893-6080(05)80131-5.

[17] Nikolay Dimitrov, Mark C. Kelly, Andrea Vignaroli, and Jacob Berg. From wind to loads: Wind turbine site-specific load estimation with surrogate models

trained on high-fidelity load databases. *Wind Energy Science*, 3(2):767–790, 2018. ISSN 2366-7443. doi: 10.5194/wes-3-767-2018.

[18] Laura Schröder, Nikolay Krasimirov Dimitrov, David Robert Verelst, and John Aasted Sørensen. Wind turbine site-specific load estimation using artificial neural networks calibrated by means of high-fidelity load simulations. In *Journal of Physics: Conference Series*, volume 1037, page 062027, 2018. doi: 10.1088/1742-6596/1037/6/062027.

[19] Nikolay Dimitrov. Surrogate models for parameterized representation of wake-induced loads in wind farms. *Wind Energy*, 22(10):1371–1389, 2019. ISSN 1095-4244, 1099-1824. doi: 10.1002/we.2362.

[20] Luis David Avendaño-Valencia, Imad Abdallah, and Eleni Chatzi. Virtual fatigue diagnostics of wake-affected wind turbine via Gaussian Process Regression. *Renewable Energy*, 170:539–561, 2021. ISSN 0960-1481. doi: 10.1016/j.renene.2021.02.003.

[21] Kelsey Shaler, John Jasa, and Garrett E. Barter. Efficient Loads Surrogates for Waked Turbines in an Array. *Journal of Physics: Conference Series*, 2265(3):032095, 2022. ISSN 1742-6596. doi: 10.1088/1742-6596/2265/3/032095.

[22] Abraham Nispel, Stephen Ekwaro-Osire, João Paulo Dias, and Americo Cunha. Probabilistic Design and Uncertainty Quantification of the Structure of a Monopile Offshore Wind Turbine. In *Volume 13: Safety Engineering, Risk, and Reliability Analysis*, page V013T13A021, Salt Lake City, Utah, USA, 2019. American Society of Mechanical Engineers. ISBN 978-0-7918-8350-1. doi: 10.1115/IMECE2019-11862.

[23] L M M van den Bos, B Sanderse, L Blonk, W A A M Bierbooms, and G J W van Bussel. Efficient ultimate load estimation for offshore wind turbines using interpolating surrogate models. In *Journal of Physics: Conference Series*, volume 1037, page 062017, 2018. doi: 10.1088/1742-6596/1037/6/062017.

[24] Mikkel Slot Nielsen and Victor Rohde. A surrogate model for estimating extreme tower loads on wind turbines based on random forest proximities. *Journal of Applied Statistics*, 49(2):485–497, 2022. ISSN 0266-4763. doi: 10.1080/02664763.2020.1815675.

[25] D. Singh, R. P. Dwight, K. Laugesen, L. Beaudet, and A. Viré. Probabilistic surrogate modeling of offshore wind-turbine loads with chained Gaussian processes. *Journal of Physics: Conference Series*, 2265(3):032070, 2022. ISSN 1742-6596. doi: 10.1088/1742-6596/2265/3/032070.

[26] Edward J. Ransley, Scott A. Brown, Emma C. Edwards, Tom Tosdevin, Kieran Monk, Alastair M. Reynolds, Deborah Greaves, and Martyn R. Hann. Real-Time Hybrid Testing of a Floating Offshore Wind Turbine Using a Surrogate-Based Aerodynamic Emulator. *ASME Open Journal of Engineering*, 2(021017), 2023. ISSN 2770-3495. doi: 10.1115/1.4056963.

[27] Manuel Fluck and Curran Crawford. A fast stochastic solution method for the Blade Element Momentum equations for long-term load assessment. *Wind Energy*, 21(2):115–128, 2018.

[28] Manuel Fluck. *Stochastic Methods for Unsteady Aerodynamic Analysis of Wings and Wind Turbine Blades*. PhD thesis, 2017.

[29] Rad Haghi and Curran Crawford. Surrogate models for the blade element momentum aerodynamic model using non-intrusive polynomial chaos expansions. *Wind Energy Science*, 7(3):1289–1304, 2022. ISSN 2366-7443. doi: 10.5194/wes-7-1289-2022.

[30] Paul S Veers. Three-dimensional wind simulation. Technical report, Sandia National Labs., Albuquerque, NM (USA), 1988.

[31] Louise Wright and Stuart Davidson. How to tell the difference between a model and a digital twin. *Advanced Modeling and Simulation in Engineering Sciences*, 7(1):13, 2020. ISSN 2213-7467. doi: 10.1186/s40323-020-00147-4.

[32] Itxaro Errandonea, Sergio Beltrán, and Saioa Arrizabalaga. Digital Twin for maintenance: A literature review. *Computers in Industry*, 123:103316, 2020. ISSN 0166-3615. doi: 10.1016/j.compind.2020.103316.

[33] Zhao Song, Christoph M. Hackl, Abhinav Anand, Andre Thommessen, Jonas Petzschmann, Omar Kamel, Robert Braunbehrens, Anton Kaifel, Christian Roos, and Stefan Hauptmann. Digital Twins for the Future Power System: An Overview and a Future Perspective. *Sustainability*, 15(6):5259, 2023. ISSN 2071-1050. doi: 10.3390/su15065259.

[34] Jeroen D. M. De Kooning, Kurt Stockman, Jeroen De Maeyer, Antonio Jarquin-Laguna, and Lieven Vandevelde. Digital Twins for Wind Energy Conversion Systems: A Literature Review of Potential Modelling Techniques Focused on Model Fidelity and Computational Load. *Processes*, 9(12):2224, 2021. ISSN 2227-9717. doi: 10.3390/pr9122224.

[35] Muhammad Fahim, Vishal Sharma, Tuan-Vu Cao, Berk Canberk, and Trung Q. Duong. Machine Learning-Based Digital Twin for Predictive Modeling in Wind Turbines. *IEEE Access*, 10:14184–14194, 2022. ISSN 2169-3536. doi: 10.1109/ACCESS.2022.3147602.

[36] Mingming Song, Babak Moaveni, Hamed Ebrahimian, Eric Hines, and Anela Bajric. Joint parameter-input estimation for digital twinning of the Block Island wind turbine using output-only measurements. *Mechanical Systems and Signal Processing*, 198:110425, 2023. ISSN 0888-3270. doi: 10.1016/j.ymssp.2023.110425.

[37] Emmanuel Branlard, Jason Jonkman, Scott Dana, and Paula Doubrawa. A digital twin based on OpenFAST linearizations for real-time load and fatigue estimation of land-based turbines. *Journal of Physics: Conference Series*, 1618 (2):022030, 2020. ISSN 1742-6596. doi: 10.1088/1742-6596/1618/2/022030.

[38] Emmanuel Branlard, Jason Jonkman, Cameron Brown, and Jiatian Zang. A digital-twin solution for floating offshore wind turbines validated using a full-scale prototype. *Wind Energy Science Discussions*, pages 1–34, 2023. doi: 10.5194/wes-2023-50.

[39] Laura Schröder, Nikolay Krasimirov Dimitrov, David Robert Verelst, and John Aasted Sørensen. Using Transfer Learning to Build Physics-Informed Machine Learning Models for Improved Wind Farm Monitoring. *Energies*, 15(2):558, 2022. ISSN 1996-1073. doi: 10.3390/en15020558.

[40] Laura Schröder, Nikolay Krasimirov Dimitrov, and David Robert Verelst. A surrogate model approach for associating wind farm load variations with turbine failures. *Wind Energy Science*, 5(3):1007–1022, 2020. ISSN 2366-7443. doi: 10.5194/wes-5-1007-2020.

[41] Charilaos Mylonas, Imad Abdallah, and Eleni Chatzi. Conditional variational autoencoders for probabilistic wind turbine blade fatigue estimation using Supervisory, Control, and Data Acquisition data. *Wind Energy*, 24(10):1122–1139, 2021. ISSN 1099-1824. doi: 10.1002/we.2621.

[42] Nikolay Dimitrov and Tuhfe Göçmen. Virtual sensors for wind turbines with machine learning-based time series models. *Wind Energy*, 25(9):1626–1645, 2022. ISSN 1099-1824. doi: 10.1002/we.2762.

[43] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[44] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling, 2018.

[45] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation, 2015.

[46] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: A review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019. ISSN 1384-5810, 1573-756X. doi: 10.1007/s10618-019-00619-1.

[47] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, 2015.

[48] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015. ISSN 1476-4687. doi: 10.1038/nature14539.

[49] Paul Westermann, Matthias Welzel, and Ralph Evins. Using a deep temporal convolutional network as a building energy surrogate model that spans multiple climate zones. *Applied Energy*, 278:115563, 2020. ISSN 0306-2619. doi: 10.1016/j.apenergy.2020.115563.

[50] Il'ya Meerovich Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4):784–802, 1967.

[51] Sergei Kucherenko, Daniel Albrecht, and Andrea Saltelli. Exploring multi-dimensional spaces: A comparison of latin hypercube and quasi monte carlo sampling techniques. 2015.

[52] Marissa Renardy, Louis R. Joslyn, Jess A. Millar, and Denise E. Kirschner. To Sobol or not to Sobol? The effects of sampling schemes in systems biology applications. *Mathematical biosciences*, 337:108593, 2021. ISSN 0025-5564. doi: 10.1016/j.mbs.2021.108593.

[53] Bonnie J Jonkman. TurbSim user's guide: Version 1.50. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2009.

[54] Jason Jonkman, Sandy Butterfield, Walter Musial, and George Scott. Definition of a 5-MW reference wind turbine for offshore system development. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2009.

[55] Bonnie Jonkman et al. Openfast/openfast: Openfast v3.3.0. `https://doi.org/10.5281/zenodo.7262094`, 2022.

[56] Kenneth Thomsen. *The Statistical Variation of Wind Turbine Fatigue Loads.* Number 1063 in Risø-R. Risø National Laboratory, Roskilde, 1998. ISBN 978-87-550-2410-6 978-87-550-2411-3. URL `https://www.osti.gov/etdeweb/servlets/purl/292704`.

[57] Henrik Stiesdal. Rotor loadings on the BONUS 450 kW turbine. *Journal of Wind Engineering and Industrial Aerodynamics*, 39(1):303–315, 1992. ISSN 0167-6105. doi: 10.1016/0167-6105(92)90555-O.

[58] B. Sanderse, S.p. van der Pijl, and B. Koren. Review of computational fluid dynamics for wind turbine wake aerodynamics. *Wind Energy*, 14(7):799–819, 2011. ISSN 1099-1824. doi: 10.1002/we.458.

[59] Tuhfe Göçmen, Paul van der Laan, Pierre-Elouan Réthoré, Alfredo Peña Diaz, Gunner Chr. Larsen, and Søren Ott. Wind turbine wake models developed at the technical university of Denmark: A review. *Renewable and Sustainable Energy Reviews*, 60:752–769, 2016. ISSN 1364-0321. doi: 10.1016/j.rser.2016.01.113.

[60] Michael Mc William, Nicolas Bonfils, Nikolay Dimitrov, and Suguang Dou. Wind farm parameterization and turbulent wind box generation. Report, DTU ; IFPEN, 2022. URL `https://ifp.hal.science/hal-04033050`.

[61] Frédéric Blondel and Marie Cathelain. An alternative form of the super-Gaussian wind turbine wake model. *Wind Energy Science*, 5(3):1225–1236, 2020. ISSN 2366-7443. doi: 10.5194/wes-5-1225-2020.

[62] Takeshi Ishihara and Guo-Wei Qian. A new Gaussian-based analytical wake model for wind turbines considering ambient turbulence intensities and thrust coefficient effects. *Journal of Wind Engineering and Industrial Aerodynamics*, 177:275–292, 2018. ISSN 0167-6105. doi: 10.1016/j.jweia.2018.04.010.

[63] Majid Bastankhah and Fernando Porté-Agel. A new analytical model for wind-turbine wakes. *Renewable Energy*, 70:116–123, 2014. ISSN 0960-1481. doi: 10.1016/j.renene.2014.01.002.

[64] Art B. Owen. On dropping the first Sobol' point, 2021.

[65] Emmanuel Branlard. pyfast. `https://github.com/OpenFAST/python-toolbox`, 2023.

[66] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186, Heidelberg, 2010. Physica-Verlag HD. ISBN 978-3-7908-2604-3. doi: 10.1007/978-3-7908-2604-3_16.

[67] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping From Saddle Points — Online Stochastic Gradient for Tensor Decomposition, 2015.

[68] Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. URL `https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/45166.pdf`.

[69] Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pages 117–122, 2018. doi: 10.1109/IIPHDW.2018.8388338.

Chapter 4

Wind Turbine damage equivalent load assessment using Gaussian process regression combining measurement and synthetic data

This chapter was first published in the MDPI Energies journal as:

Haghi R, Stagg C, Crawford C. Wind Turbine Damage Equivalent Load Assessment Using Gaussian Process Regression Combining Measurement and Synthetic Data. Energies. 2024; 17(2):346. https://doi.org/10.3390/en17020346

Author Contributions

RH and CS developed the idea under the supervision of CC. RH built and processed the hybrid and SCADA databases. CS and RH developed the necessary computer code. RH wrote the paper in consultation with and under the supervision of CC.

Abstract

Assessing the structural health of operational wind turbines is crucial, given their exposure to harsh environments and the resultant impact on longevity and performance. However, this is hindered by the lack of data in commercial machines and accurate models based on manufacturers' proprietary design data. To overcome these

challenges, this study focuses on using GPR to evaluate the loads in wind turbines using a hybrid approach. The methodology involves constructing a hybrid database of aero-servo-elastic simulations, integrating publicly available wind turbine models, tools and SCADA measurement data. Then, constructing GPR models with hybrid data, the prediction is validated against the hybrid and SCADA measurements. The results, derived from a year of SCADA data, demonstrate the GPR model's effectiveness in interpreting and predicting turbine performance metrics. The findings of this study underscore the potential of GPR for the health and reliability assessment and management of wind turbine systems.

4.1 Introduction

Wind turbines operate in harsh environments subjected to intense cyclic high- and low-frequency forces that can compromise their longevity and overall performance. Assessing the fatigue of these structures is crucial for ensuring their optimal operation and maintenance scheduling, determining their remaining useful lifetime, and considering potential lifetime extensions within windfarm [1, 2]. The assessment of fatigue damage accumulation in all components, based on measurements using sensors on a wind turbine, offers a solution. However, this approach requires a large array of sensors, leading to both logistical and financial constraints that are typically not employed on commercial machines [3, 4, 5]. The use of aero-servo-elastic simulators has been proposed as an alternative solution. These simulators can generate vast amounts of data, which engineers and researchers can subsequently analyze to assess the fatigue life of wind turbines. However, while these simulations provide valuable insights, it is essential to recognize that they often do not align completely with a turbine's actual environmental conditions and specific as-built characteristics. Thus, while simulators are a valuable tool, their results should be interpreted cautiously and supplemented with real-world data wherever possible to ensure accurate fatigue life assessments. Modern large wind turbines are equipped with Supervisory Control and Data Acquisition (SCADA) [6]. SCADA systems typically collect over 200 variables, often recording and storing as 5- or 10-minute averages along, with basic statistics such as minimum (min), maximum (max), and STD for each interval [7]. However, SCADA has reliability and accuracy issues and typically does not include any data field directly related to loads [8]. When load measurements are available from the SCADA system, they cannot be used directly for calculating the fatigue load. This is

because the 10-minute time scale is insufficient to capture a wind turbine's dynamic behaviour, which mainly requires high-resolution time series. Given all these factors and considerations, one potential solution is to use the SCADA environmental measurements as input for an aero-servo-elastic simulator to compute structural loads, in the absence of physical, e.g. strain gauge measurements. The output can then be utilized to construct a data-driven model capable of estimating and indicating DEL on turbine components.

Load assessment of a wind turbine is a complex task whether it be in the design phase for a machine, layout optimization of a wind farm, or backing out from operational data. Therefore, many attempts have been made to simplify this task in the literature [9, 10, 11, 12, 13]. Dimitrov et al. discussed five different methods, including Kriging and PCE, for load assessment using synthetic data [14]. Their results indicate that the mean wind speed and turbulence intensity have the most significant effect on fatigue load estimation. In another study, Haghi and Crawford developed a PCE to map the random phases of synthetic wind to the loads on a wind turbine rotor [15]. Most of the studies in the literature focus on the realm of SMs, attempting to map wind speed by itself or combined with other relative variables to the fatigue or extreme loads of a wind turbine. In recent years, with the rapid growth of ML methods, there has been a shift towards using these methods for load assessment. Dimitrov et al. developed an ANN to map different environmental conditions to the DEL on a turbine [14]. Schröder et al. used ANN to develop a surrogate model capable of predicting the fatigue life of a wind turbine in a wind farm, considering changes in loads [16]. More recently, Dimitrov and Göçmen developed a virtual sensor based on a sequential ML method that can provide load time series for different components of a wind turbine [17].

Condition Monitoring (CM) for wind turbines is an activity that monitors the state of the turbine [18]. CM is vital for wind turbines as it can reduce downtime, failure, and maintenance costs. There are various techniques available for CM of wind turbines; however, many are either expensive or complex [3, 8]. Consequently, utilizing data from SCADA for CM is appealing, as these data are available for the majority of turbines and do not incur additional costs [19]. Tautz-Weinert and Watson provides an extensive review of the different CM methods that utilize data from SCADA. A few of these methods address damage modelling and fatigue of com-

ponents. The concept behind damage modelling is to integrate measurements from SCADA with a physical model to understand better damage progression [8]. Gray and Watson introduced a probability of failure methodology incorporating relatively simple failure models and successfully tested this method on a wind farm with a high rate of gearbox failure [20]. Galinos et al. created a map of the fatigue life distribution for the Horns Rev 1 offshore wind farm turbines using SCADA wind speed measurements and aeroelastic simulations [21]. Alvarez and Ribaric used SCADA to describe the wind turbine torque histogram and introduced a methodology for a physics-based gearbox fatigue failure prediction [22]. Remigius and Natarajan utilized SCADA measurements to estimate the wind turbine main shaft using an inverse problem-based approach [23]. The examples mentioned above are mainly based on a physics-driven approach.

In recent years, with the emergence of data-driven methods, the integration of SCADA measurements with ML-based methods has become more popular among researchers. Pandit et al. provided an extensive review of data-driven CM approaches [24]. More specifically, data-driven methods using SCADA measurements have been increasingly adopted for predicting fatigue life and damage. Vera-Tudela and Kühn employed ANN to map wind farm varying flow conditions to fatigue loads and demonstrated the robustness of this method by using data from two distinct wind farms [25]. Natarajan and Bergami found that an ANN could predict turbulence and loads on the blade and tower by considering rotor speed, power production, and blade pitch angle from SCADA measurements, validating the loads using an instrumented turbine [26]. Mylonas et al. developed a regenerative model based on a convolutional variational autoencoder, capable of predicting DEL on a wind turbine blade root and the uncertainty of loads using only 10-minute average SCADA data [27].

A Gaussian Process (GP) is a type of ML technique used for both regression and classification problems. It is a data-driven, non-parametric method that does not rely on a specific functional form. Instead, it focuses on a distribution of functions that align with the data it is analyzing [28]. The application of GP in wind turbine research and engineering has grown due to the ease of implementation, versatility, and adaptability of the method, as well as its ability to provide uncertainty estimates. For instance, Pandit and Infield utilized GPR to capture failures due to yaw misalignment using SCADA [29]. Li et al. employed Gaussian Process Classification (GPC) to

detect and predict wind turbine faults from SCADA data, where the provided probabilistic knowledge aids in maintenance management [30]. Herp et al. utilized GP to forecast wind turbine bearing failure a month in advance based on wind turbine bearing temperature residuals [31]. Avendaño-Valencia et al. predicted wind turbine loads in downstream wakes, calibrating a GPR based on local or remote wind or load measurements [32]. Wilkie and Galasso assessed the fatigue calculation reliability of offshore wind turbines using a GPR, where inputs consisted of site-specific environmental conditions and turbine structural dynamics [33]. Singh et al. employed chained GP to derive the probability distribution function of offshore wind turbine loads based on stochastic synthetic loads [34].

4.1.1 Motivation

In this manuscript, we aim to create a simple yet dependable probabilistic model for predicting damage using limited SCADA measurements and utilizing publicly available wind turbine models. The inflow turbulence is stochastic, leading to load responses that are aptly represented as random variables. The influence of these unpredictable factors on loads heavily depends on average environmental conditions and their variance. This variance in load response, known as *heteroscedasticity* in statistical terms, implies that at lower wind speeds, the inflow turbulence has a lesser impact on load variability compared to higher wind speeds [34]. Heteroscedasticity directly affects the DEL of a wind turbine. The challenge is that to obtain an accurate distribution of the DELs for an operational turbine, we require many data points. Ideally, this could be achieved with an extensive array of sensors on wind turbines, which is not feasible. One approach involves running simulations to enrich the database and attain improved distributions. However, two primary challenges exist: a) the models lack accuracy, and b) wind turbine manufacturing companies view models as their intellectual property, making them generally inaccessible.

Given these challenges, our proposal is not for a highly accurate model to predict the DEL down to the minutest details. Instead, we advocate for a straightforward methodology to offer a probabilistic model built on hybrid SCADA and publicly available turbine models. This model can approximate the DEL distribution at each wind speed and demonstrate the trend of the DEL distribution as wind speed varies.

Although this model might not be precise enough to indicate the remaining useful lifetime with high accuracy, it can roughly gauge the turbine fatigue health condition and relative damage of machines within a wind farm. Such a model can serve as a quick indicator to pinpoint turbines at risk, warranting further investigation. Additionally, it can assist in reducing the uncertainty of a turbine's health condition for financial and banking purposes. This model demonstrates benefits for "asset reliability" and "asset health," especially in mitigating investor risks when considering the purchase of operational wind farms.

4.1.2 Objective

For this research, we had access to a year's worth of data from an undisclosed turbine in an undisclosed onshore wind farm's SCADA system. The objectives of this manuscript are as follows:

- Create a database of synthetic DEL based on publicly available turbine models with SCADA wind measurements as input.
- Develop a probabilistic model based on the database that represents the distribution of the descriptive statistics and DEL at varying wind speeds.
- Validate the probabilistic model by contrasting its output with the limited available measurements.

4.1.3 Paper outline

The paper is organized as follows: Section 4.2 starts with an overview of the methodology, depicted in Figure 4.1, and continues with a description of the SCADA system used in the study, including data collection and processing methods. This is followed by an explanation of joint distributions and sampling in Section 4.2.2, and the basics of aero-servo-elastic simulations and their postprocessing. Section 4.2.5 introduces the Gaussian Process Regression (GPR) methodology applied in this study and concludes Section 4.2 with a definition of the error metrics used. Section 4.3 begins with the conditions under which results are extracted and GPR models are trained, continuing with the validation of these models against empirical data. The accuracy of the trained GPR, using hybrid data, is compared with both simulations and SCADA data

in Sections 4.3.1 and 4.3.6. Section 4.3 ends with proposed practical applications for the developed model. The manuscript concludes with Section 4.4, summarizing the main findings and suggesting future research in using GPRs for wind turbine primary health assessment.

4.2 Methods

This section presents the methodology used to construct a GPR model using hybrid simulation and validate the GPR predictions against both hybrid simulation and SCADA measurements. Figure 4.1 provides an overview of our approach, consisting of four blocks. The arrows illustrate the data flow between these blocks, databases, and processes. Hereafter, "wind speed" refers to the measured wind speed from SCADA, unless otherwise specified. It is worth mentioning that the wind speed sensor on a wind turbine is typically placed on top of the nacelle behind the rotor, and its readings differ from the true inflow wind speed.

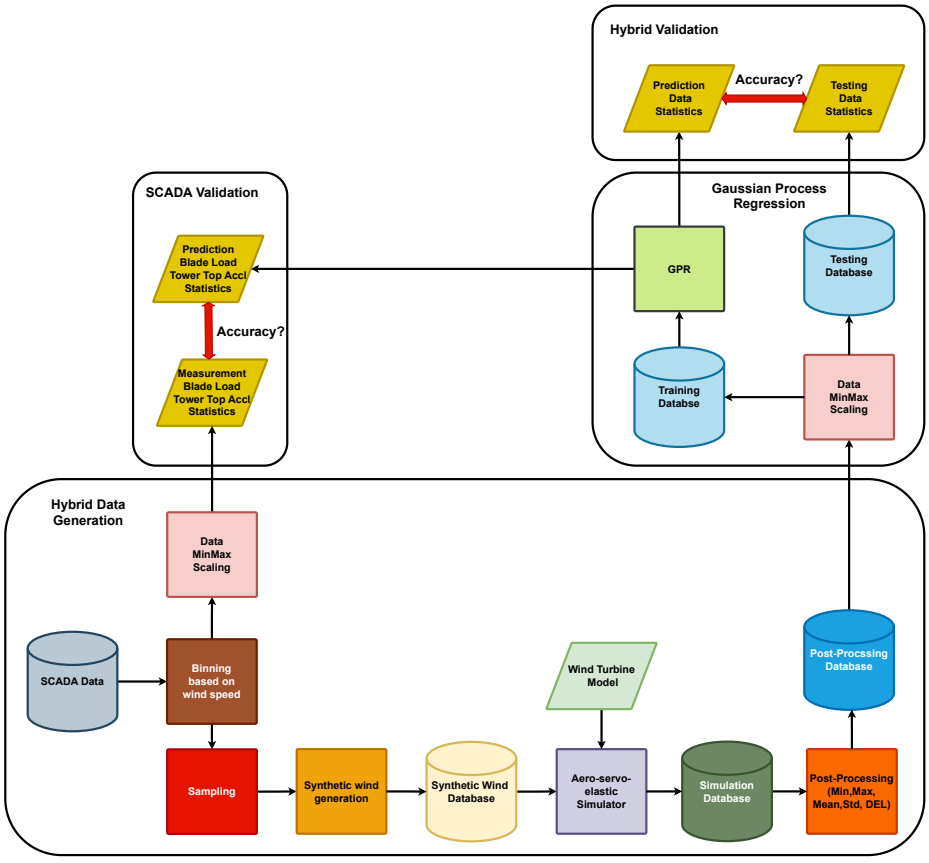


Figure 4.1: The methodology employed in this manuscript

The hybrid simulations generation block demonstrates the procedure for generating hybrid simulation data from SCADA measurements. Termed "hybrid," this data combines SCADA measurements with synthetic data generation to create a comprehensive database. The SCADA data is binned to a resolution of $1m/s$, spanning the cut-in and cut-out wind speeds. The turbine in this study operates between $3m/s$ and $25m/s$. For each bin, we establish a joint distribution of mean wind speed and STD of wind speed. The mean wind speed adheres to a uniform distribution between the bin's upper and lower bounds, whereas the wind speed STD distribution sampling per bin is tested on both Weibull and uniform distributions separately.

Subsequently, using Sobol sampling, n samples are drawn from this joint distribution for each wind speed. Each sample, comprising a mean and STD of wind speed, generates a synthetic wind field. This results in a corresponding synthetic wind field per sample. These synthetic wind fields and the wind turbine model are inputs to the aero-servo-elastic simulator. The simulator outputs load time series for various components of the wind turbine model. These load time series are post-processed to extract statistical descriptions (minimum, maximum, mean, and STD) and the DEL for each component. In this manuscript, the post-processed outputs are termed QoI. The QoI are stored in a database, referred to as the post-processing database.

The Gaussian Process Regression block outlines the technique employed to build the GPR model using hybrid simulation output. The post-processing database is initially scaled using the MinMax method to normalize all data fields between 0 and 1. The database is then divided into two non-overlapping datasets: Training and Testing. A separate GPR model is trained for each QoI for each wind turbine component load. The GPR model outputs the mean and STD of the QoI at each wind speed, representing the prediction data statistics. The testing data similarly provides the mean and STD of the QoI at each wind speed, known as the testing data statistics.

Figure 4.1 illustrates two validation blocks: Hybrid Validation and SCADA Validation. For Hybrid Validation, the testing data statistics are compared against the prediction data statistics. If the SCADA data includes tower top acceleration or blade load statistics, the GPR prediction distribution for these is also validated.

In the following sections, we delve into the processes and steps depicted in Figure

4.1 in greater detail.

4.2.1 Supervisory Control and Data Acquisition measurement, binning and scaling

The comprehensiveness of the collected SCADA data offers a wide range of insights for system analysis. Notably, the SCADA data encompasses numerous data fields, with our primary interest in wind speed statistics for hybrid simulation database generation, generated power for wind turbine model validation, and, if available, tower top acceleration and blade load statistics for measurement validation.

We binned the SCADA data based on mean wind speed to gain a broader perspective on wind turbine operation through the wind speed statistics. The bin center corresponds to an integer wind speed value, with the upper and lower bounds set at $\pm 0.5 m/s$ of that value. For each bin, we calculated the mean of the measured mean wind speed, the mean of the STD of the wind speed, and the STD of the STD of the wind speed, resulting in three wind speed statistics for each bin.

Min-max scaling is a common preprocessing step in data analysis and machine learning, involving transforming features to a specified range, often $[0, 1]$ [35]. According to Rasmussen and Williams, scaling the data is recommended for GPR to ensure numerical stability [28]. For the post-processing database, we scaled the data based on the minimum and maximum values in each wind turbine channel output, effectively constraining each scaled post-processed output to the $[0, 1]$ range. Additionally, if SCADA loads or acceleration measurements were available, they were scaled similarly to facilitate comparison with the GPR output. Furthermore, we adopted min-max scaling for all output data in compliance with confidentiality requirements.

4.2.2 Joint distributions and sampling

Hybrid simulation data generation aims to build a comprehensive database of the loads on a wind turbine, closely resembling real-world conditions. This process, known as data assimilation [36], merges observational data (in our case, SCADA data) with

model predictions to produce a more complete estimate of the current state of the system and future evolution. One approach to account for measurement uncertainties is to define them as random variables with specific distributions. We utilized wind speed statistics extracted in Section 4.2.1 to construct joint distributions of mean wind speed and STD of wind speed for each bin. A uniform distribution was defined to cover the full range of the bin for the mean wind speed equally. Regarding the STD, we opted to test two alternatives: a) fitting a Weibull distribution to the STD of wind speed, and b) applying a uniform distribution to the STD, with bounds set at the minimum and maximum values per bin. The assumption of a uniform distribution for wind speed has been previously established in literature [14, 16]. The choice of a Weibull distribution for the STD is based on data observations, and a uniform distribution is selected to encompass all possibilities, particularly when SCADA measurements in a bin are unavailable.

We employed the QMC Sobol sampling technique, as detailed in [37]. This method is preferred in our study for its reliability and computational efficiency, as noted in [38]. Sobol's technique is repeatable and ensures enhanced uniformity across sampled distributions, a feature emphasized in [39]. Hereinafter, "the sample" refers to a two-data-point vector comprising mean wind speed and STD of wind speed. We took n samples per bin, resulting in n unique samples for m bins, which are then used to generate $n \times m$ synthetic wind time histories.

4.2.3 Synthetic wind generation, wind turbine models and aero-servo-elastic simulations

To perform aero-servo-elastic simulations, we require synthetic wind time histories that closely resemble real-world wind conditions experienced by the turbine. To achieve this, we constructed joint distributions from the SCADA measurements and calculated statistics for each wind speed bin as explained in Section 4.2.2. The objective was to generate synthetic wind time histories that faithfully replicate the actual wind conditions corresponding to the mean and STD of wind samples. To do so, we used the samples' wind speed and STD as the input to TurbSim [40]. The output of TurbSim is a "full-field" wind time history in TurbSim format. TurbSim, a synthetic wind generator, produces wind time histories with both spatial and temporal components for aero-servo-elastic simulators. A comprehensive explanation of this synthetic

wind generator can be found in [40].

The resulting $n \times m$ full-field synthetic wind time series are stored in the synthetic wind database. The full-field synthetic wind data serves as the environmental input for our aero-servo-elastic simulator. These simulations require synthetic wind time series and the integration of aerodynamic, aeroelastic, and controller models. Each wind turbine model encompasses modules for aerodynamics, aeroelasticity, and control. To conduct these simulations, we employed OpenFAST, an aero-servo-elastic solver developed by NREL [41]. The output from OpenFAST provides detailed load information for various wind turbine components, including blades, towers, and gear systems, spanning both time and space. Our simulations adhere to the IEC standards for energy production under DLC 1.2, as specified in the IEC standards [42].

The provided SCADA data corresponds to a year's worth of data, which includes loads and acceleration data. However, as the turbine models are the intellectual property of the wind turbine manufacturers, we did not have access to them. Therefore, we opted for the NREL 5MW turbine for the aero-servo-elastic simulations [43] as this model is well established in the literature, demonstrates robustness against fluctuations in wind speed seed during simulations, the controller is well defined, and the rotor size is comparable with the turbine we have access to the SCADA measurements. Moreover, our tests indicate that the NREL 5MW model provides simulation results most similar to SCADA data at hand compared to other publicly available turbines [44, 45, 46].

4.2.4 Post-processing database

We have compiled a comprehensive database incorporating all time series data from the simulation outputs. Following its creation, the data underwent post-processing to derive simulation QoI, namely descriptive statistics and DEL for assessing loads and fatigue. Additionally, DEL computation adheres to the Palmgren–Miner linear damage rule, as elaborated in [47] and [48]. The DEL can be expressed as follows:

$$DEL = \left(\frac{\Sigma n_i R_i^m}{n_{eq}} \right)^{1/m} \tag{4.1}$$

Here, m represents the Wöhler slope, while R_i and n_i pertain to load ranges and the

corresponding number of cycles, respectively. The DEL outcome is derived through rainflow counting of the load time series [49, 47]. n_{eq} denotes the equivalent number of load cycles, typically equivalent to the length of the simulations in seconds. The post-processing database encompassed all calculated descriptive statistics and DELs from every simulation output.

Subsequently, the post-processing database is subjected to MinMax scaling and binning which was explained in Section 4.2.1.

4.2.5 Gaussian Process Regression

GPR is a non-parametric Bayesian method widely used for regression tasks [28]. At the core of GPR lies its assumption that observed target values follow a multivariate Gaussian distribution. One of its notable features is its ability to provide probabilistic predictions, offering both mean and variance functions to quantify prediction uncertainty. Given a dataset $(x_1, y_1), \ldots, (x_N, y_N)$, a Gaussian Process defines a distribution over functions characterized by a mean function $m(x)$ and a covariance (kernel) function $k(x, x')$. The GPR can be expressed as:

$$f(x) \sim \mathcal{GPR}(m(x), k(x, x')) \tag{4.2}$$

where, $m(x)$ represents the mean function, and $k(x, x')$ is the covariance function, capturing the data point relationships. Predictions f_* at new data points x_* are expressed as predictive mean $\boldsymbol{\mu}_*$ and variance $\boldsymbol{\Sigma}_*$:

$$\boldsymbol{\mu}_* = \mathbf{K}_*^T(\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}y \tag{4.3}$$

$$\boldsymbol{\Sigma}_* = \mathbf{K}_{**} - \mathbf{K}_*^T(\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}\mathbf{K}_* \tag{4.4}$$

where, $\mathbf{K}$ is the covariance matrix for the training data, $\mathbf{K}_*$ represents the covariance matrix between training and test data, and $\mathbf{K}_{**}$ is the covariance matrix for the test data. y is the vector of training targets, and σ_n^2 is the noise variance. Typically, noise variance and other hyperparameters of GPR and kernel function parameters are estimated from the data, often using techniques like maximum likelihood estimation. The log-likelihood of observations conditioned on hyperparameters is expressed as:

$$\log p(y|x,\theta) = -\frac{1}{2}y^T(\mathbf{K}+\sigma_n^2\mathbf{I})^{-1}y - \frac{1}{2}\log|\mathbf{K}+\sigma_n^2\mathbf{I}| - \frac{N}{2}\log 2\pi \tag{4.5}$$

where, y represents the vector of observed target values, $\mathbf{K}$ is the covariance matrix calculated using the kernel function for training inputs, and σ_n^2 denotes the noise variance. $|\mathbf{K}+\sigma_n^2\mathbf{I}|$ signifies the determinant of the matrix, and N is the dataset size. Maximum likelihood estimation aims to determine the hyperparameters θ that maximize this log-likelihood.

For the problem at hand, the x values are the wind speeds, and the y values are the QoI. Recalling the number of bins m and the number of samples n, we have $n \times m$ data points, which can result in a large number of data. Also, the target data y is a *heteroscedastic* variable. In our case, it means the data variance is not constant across wind speeds. Due to these two characteristics of the data at hand, standard GPR formulation is ineffective for our purpose. The standard GPR method implementations require $O(n^3)$ computation where n is the number of data samples [50]. Also, standard GPR assumes the variance across the data is constant. [51]. To tackle these two, we used AGPR for the large-size dataset challenge with "inducing points" [50] and Predictive Log Likelihood (PLL) for the heteroscedasticity challenge [52]. AGPR introduces a set of inducing points or pseudo-inputs representing a subset of the training data. The GP is conditioned on these inducing points rather than the entire dataset, reducing computational complexity [50]. Then, PLL directly aims for the posterior distribution estimation [52]. Both of these methods have been thoroughly explained in various literature. For a deeper understanding of these methods, readers are encouraged to refer to [50, 53, 52]. In this work, we utilized `GPyTorch` for building the GPR models [54].

4.2.6 Measurement statistics, error metrics

If the SCADA measurement data includes loads or tower top acceleration information, it offers an opportunity to compare these data points with the AGPR output. Given that the output from AGPR is probabilistic, processing the SCADA measurement data to extract relevant statistics becomes necessary. This procedure mirrors the one detailed in Section 4.2.1. Initially, we scale the measurements using MinMax scaling. Subsequently, we categorize the scaled SCADA measurement loads and acceleration

data based on SCADA wind speed measurements and then calculate the mean and STD of acceleration and loads within each bin.

We assess the disparity between the SCADA measurement and AGPR output, or between the AGPR output and the testing datasets, using KL divergence. The KL divergence is formulated as:

$$D_{KL}(G||M) = \frac{1}{2}\left[\log\left(\frac{\sigma_M^2}{\sigma_G^2}\right) + \frac{\sigma_G^2 + (\mu_G - \mu_M)^2}{\sigma_M^2} - 1\right] \tag{4.6}$$

Where in (4.6), G represents the AGPR output with mean μ_G and variance σ_G^2, and M signifies the measurement or simulation data with mean μ_M and variance σ_M^2. This comparison involves computing the KL divergence, with the testing database and SCADA measurement serving as the reference or "ground truth." In cases where we have samples from both AGPR and SCADA, the KL divergence is formulated as:

$$D_{\mathrm{KL}}(G \parallel M) = \sum_i G(i) \log\left(\frac{G(i)}{M(i)}\right) \tag{4.7}$$

where $G(i)$ and $M(i)$ are the probability distributions at bin i for the two distributions, respectively. The bins are set identically for both distributions. The KL divergence has the minimum value of zero and no upper bound. If the KL divergence is zero, it indicates that the two distributions being compared are identical. Therefore, smaller values are preferable. For a more detailed discussion of this topic, interested readers are referred to Murphy [55].

4.3 Results and discussion

In this section, we explain the conditions used for generating the results, followed by the presentation of the results and their corresponding discussions. In this section, whenever we refer to a QoI or a data field, it is scaled using MinMax scaling.

4.3.1 SCADA measurement

As mentioned, we considered a year's worth of SCADA measurement data.The SCADA data is filtered only to include data points where the power generated exceeds zero. To understand the environmental conditions in which the turbine operates, the mea-

sured mean wind speed and STD wind speed are illustrated in Figure 4.2.

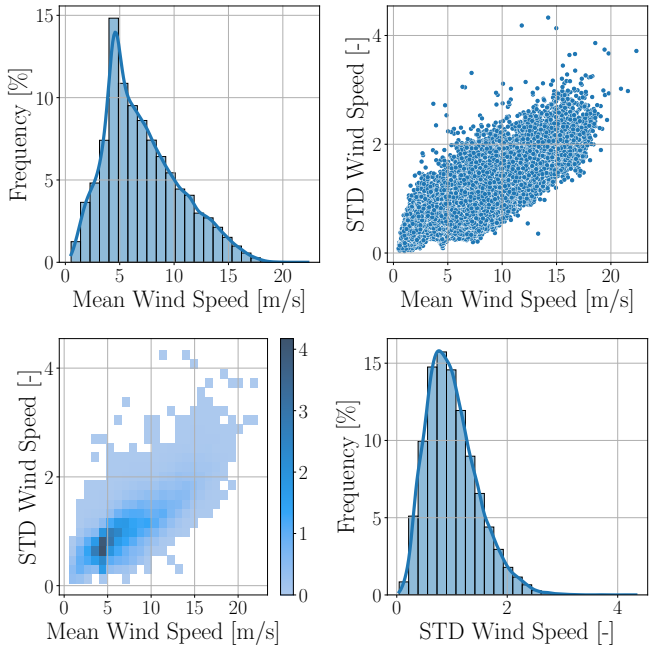


Figure 4.2: Illustration of SCADA Measurement Data: The top right panel shows raw data for mean and STD wind speeds. The diagonal panels feature histograms for both mean wind speed and STD wind speed, accordingly. The bottom left panel presents a 2D histogram, elucidating the relationship between mean wind speed and STD wind speed measurements.

In Figure 4.2, the diagonal plots are histograms of the mean wind speed and STD of wind speed. The off-diagonal plots are scatter plots for the mean and STD on the top right, and the 2D histogram of the mean and STD at the bottom left. This shows we have reasonable variability in a year to cover different operational conditions.

In addition to wind measurement, the SCADA database contains other data fields. Table 4.1 provides these additional data fields from the SCADA we utilized in this study.

We will utilize the data fields in In Table 4.1 in Sections 4.3.5 and 4.3.7 to test the simulation models and the AGPR outputs. We received the measurement data as a 10-minute average without access to the underlying time series; therefore, we cannot comment on the noise's effect or the noise level in the data.

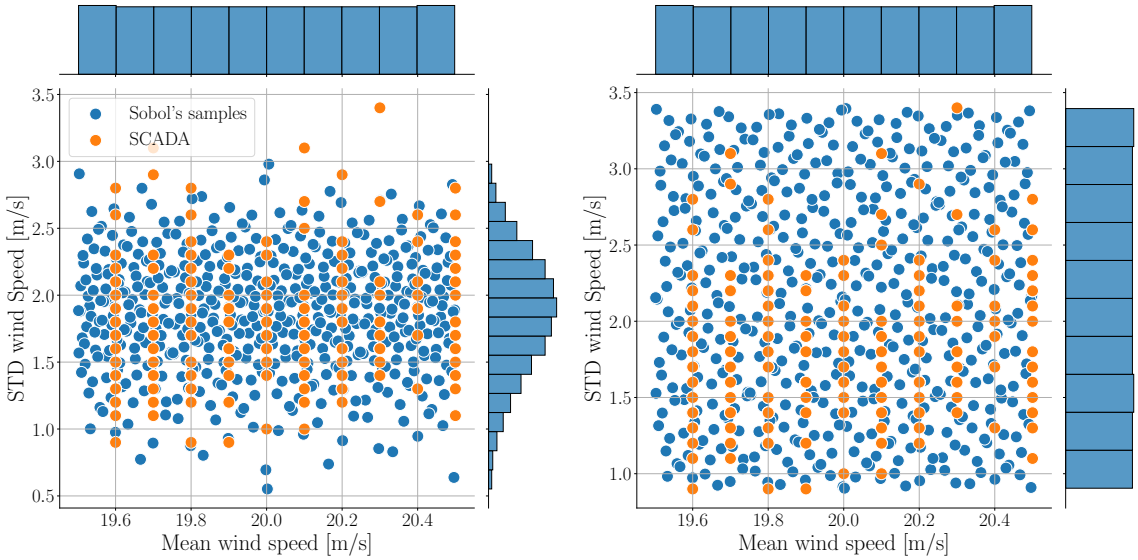
Table 4.1: The SCADA data fields used in this study. The abbreviations are TT and OFP, while Res. is short for resultant and accel. for acceleration.

Datafield	Unit	10-min avg	10-min STD	10-min min	10-min max
Power	[kW]	✓	-	✓	✓
Rotor Speed	[rpm]	✓	-	✓	✓
OFP blade load	[Nm]	✓	✓	✓	✓
TT Res. accel.	[mm/s^2]	✓	✓	✓	✓

4.3.2 Joint distributions and sampling

As described in Section 4.2.1, we binned this raw data based on wind speed and processed them to construct joint distributions within each bin. The centers of the bins range from $3m/s$ to $25m/s$ with a resolution of $1m/s$. The edges of the bins are $\pm 0.5m/s$ from the center of the bins. The edges of the uniform distribution for wind speed coincide with the bin edges. We have two types of joint distributions per bin; UW joint distribution for each bin, and UU joint distribution per bin. After building the joint distributions, we sampled from these distributions as input to the synthetic wind generator. Each sample contains a mean wind speed and standard deviation.

As detailed in Section 4.2.2, we employed the Sobol sampling technique for our investigation. To maintain the balance characteristics of Sobol sequences, they must be of the form 2^n [56]. Consequently, we chose 2^9 samples from the set distributions for each bin. We opted for this sample size as the Sobol sampling approach allows us to minimize the sample count without compromising the method's advantages or needing to resample areas. Additionally, it provides a substantial database for testing purposes. To illustrate, we presented the samples of the $20m/s$ bin for both UW and UU in Figure 4.3. In Figure 4.3, it appears the provided SCADA for the mean wind speed is rounded to the nearest $0.1m/s$, which is why they are clustered at specific wind speeds. This aspect should be considered when comparing the SCADA with AGPR outputs later in this manuscript.



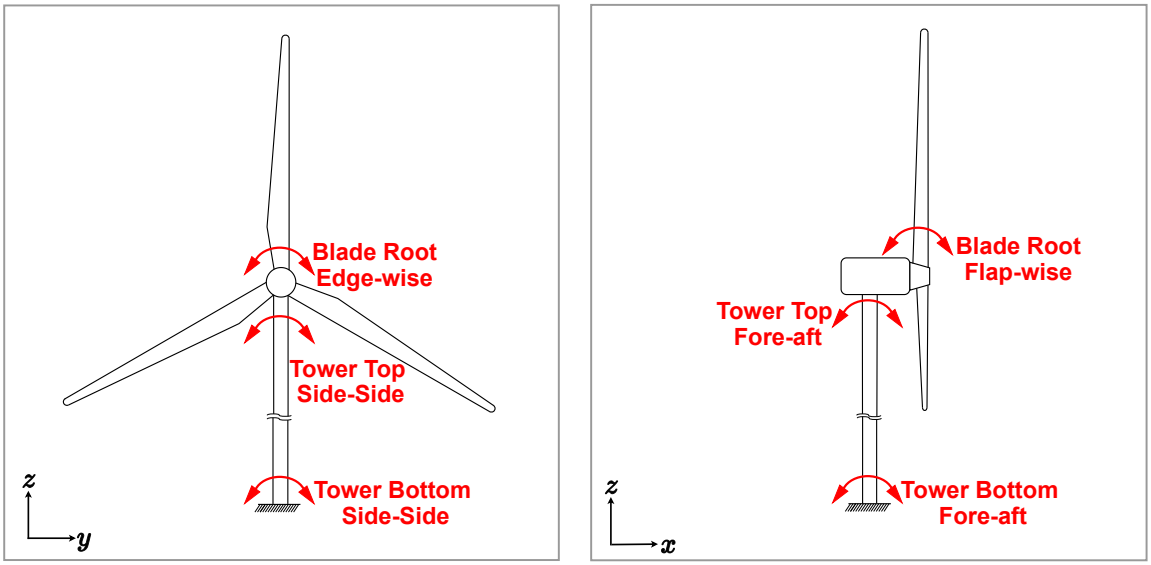
(a) UW samples

(b) UU samples

Figure 4.3: 512 Sobol's samples example for the $20m/s$ wind speed bin for UW and UU options.

4.3.3 TurbSim and OpenFAST output

The output format of TurbSim is comprehensively detailed in [40]. This output encompasses both spatial and temporal dimensions. Spatially, it provides a synthetic wind time series at various grid locations across the rotor plane. Specifically, TurbSim generates three wind time series corresponding to x, y, and z directions for each grid point. In our research, we utilized a grid of 15 in the x axis by 15 in the z axis over the rotor plane. Temporally, the synthetic wind series operates at a frequency of $20Hz$ over a span of $720sec$. As previously indicated, we executed aeroelastic simulations on the wind turbine model outlined in Section 4.2.3 via OpenFAST. OpenFAST provides various outputs, referred to as channels, for different turbine parts. Detailed descriptions of these channels are available in [57].



(a) OpenFAST moment output channels in the y, z plane

(b) OpenFAST moment output channels in the x, z plane

Figure 4.4: Figures (a) and (b) present the schematic depiction of the turbine highlighting the moment output load channel locations. The red arrows depict the moments according to OpenFAST [46].

In this study, we evaluated six distinct output channels for DEL along with the mean values for TT side-side, fore-aft, resultant absolute acceleration, rotor speed and power. These moments encompass BR in edge-wise and flap-wise directions, TT in fore-aft and side-side motions, and similarly, tower-base (TB) in fore-aft and side-side movements. The wind turbine load channel positions used in this study are graphically depicted in Figures 4.4a and 4.4b. A comparison between the OpenFAST output channel descriptors and the names we adopted for this research is provided in Table 4.2.

The out-of-plane BR moment is similar to the edge-wise BR moment in definition, with the primary difference being the coordinate system in which the moment is calculated. OpenFAST calculates the moment in the "blade" coordinate system for the edge-wise moment, while the out-of-plane moment is calculated in the "cone" coordinate system. For a detailed explanation of these coordinate systems, the interested reader is referred to the OpenFAST manual [57]. The TT resultant acceleration, YawBrTAccl, is not directly provided as an OpenFAST output but is computed separately after running the simulations. The resultant acceleration is formulated as follows:

Table 4.2: List of channel descriptions and the adopted naming and units.

OpenFAST Channel label	Adopted Name	Post-processing	Unit
GenPwr	Power Output	10-min mean	[kW]
RtSpeed	Rotor Speed	10-min mean	[kW]
RootMyc1	Out-of-plane BR moment	10-min mean	[kNm]
YawBrTAxp	TT fore-aft acceleration	10-min mean	[m/s^2]
YawBrTAyp	TT side-side acceleration	10-min mean	[m/s^2]
YawBrTAccl	TT resultant acceleration	10-min mean	[m/s^2]
RootMxb1	Edge-wise BR moment	DEL	[kNm]
RootMyb1	Flap-wise BR moment	DEL	[kNm]
YawBrMxp	TT side-side	DEL	[kNm]
YawBrMyp	TT fore-aft	DEL	[kNm]
TwrBsMxt	TB side-side	DEL	[kNm]
TwrBsMyt	TB fore-aft	DEL	[kNm]

$$YawBrTAccl(t) = \sqrt{YawBrTAxp(t)^2 + YawBrTAyp(t)^2} \tag{4.8}$$

For this analysis, we initiated a total of $2 \times 23 \times 2^9 = 23552$ aeroelastic simulations via OpenFAST. These simulations were run in 5888 parallel batches, each comprising 4 simulations, utilizing resources from the Digital Research Alliance of Canada. Each simulation lasted for $720s$, with the initial $120s$ omitted to negate start-up effects. The aeroelastic simulation used a time step of $0.005s$ for the simulations, while the output registered at $20Hz$ resolution. Upon completion of all simulations and subsequent database formation, we derived statistical descriptions and DEL for every simulation concerning the channels of interest, adopting a Wöhler slope of $m = 4$ and $n_{eq} = 600$ in Equation (4.1). For simplicity, one Wöhler slope is used for all the load channels, with n_{eq} equaling the simulation length in seconds [47, 48]. The 10-minute mean of power generation was also considered. The `pyFAST` Python library, referenced in [58], was employed to interpret the OpenFAST output files, compute statistical values, and determine the DEL. Through our testing of DEL calculation ($10Hz$,$5Hz$ and $1Hz$), we have discovered that altering the sampling frequency of simulation outputs had little impact. Furthermore, the pattern of DEL across different wind speeds remained steady for different sampling frequencies. As all outputs in this study are scaled to be between 0 and 1, the trend of DEL bears greater importance than its numerical values.

4.3.4 AGPR training and testing

For each moment or acceleration channel in Table 4.2, we developed and trained a AGPR model. The basic background of the AGPR models is explained in Section 4.2.5. The AGPR models and training settings implemented in `GPyTorch` are presented in Table 4.3.

Table 4.3: `GPyTorch` configurations for AGPR model training on DEL and 10-minute mean values.

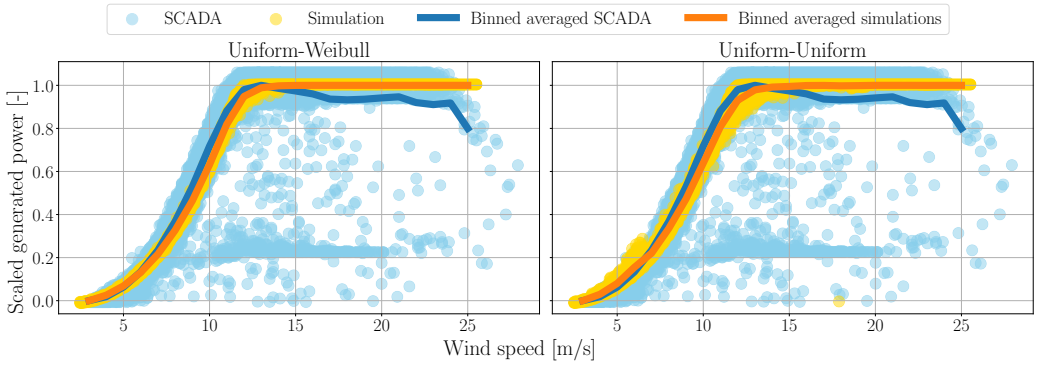
	DEL	10 min. mean	Reference
Model class	Approximate GP	Approximate GP	[50]
Kernel (length scale)	RBF Kernel (0.7)	RBF Kernel (0.7)	[28]
Marginal likelihood class	PLL	PLL	[52]
Variational distribution	Cholesky	Cholesky	[54]
Training dataset size	5888	11776	-
Number of inducing points	64	64	-
Number of iterations	500	500	-

The DEL moments database is divided into two equally sized datasets for training and testing. The training dataset is further divided into 80% for training and 20% for validation of the AGPR model. As the training dataset is large, k-fold cross-validation is not utilized, which is more applicable for smaller databases to prevent overfitting [59]. For the 10-minute mean moments and acceleration, we trained the AGPR on the entire simulation outputs dataset with 80% for training and 20% for validation split. Then, we tested the GPR on the SCADA data. How the turbine load for each output channel changes with wind speed depends on the output location. The settings and parameters listed in Table 4.3 were tuned manually. Different settings were tested for different outputs, and the configurations provided were optimized to achieve similar accuracy across all outputs.

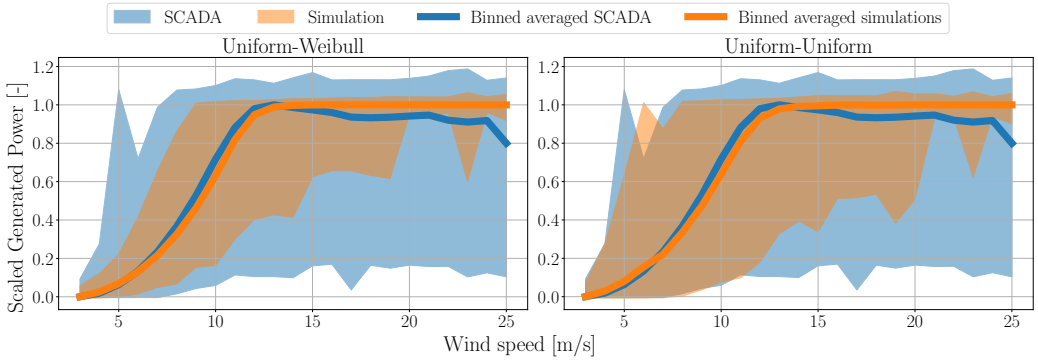
4.3.5 Wind turbine model verification

After running the simulations and extracting the QoI to construct the databases, the initial step involved verifying the simulation model's accuracy against SCADA measurements. Power output and rotor speed were two of the most reliable indicators

among the data fields available for measurements and simulations. The trends of these two variables indicate the level of agreement between the wind turbine model and simulations. Additionally, comparing the range of these two outputs for both simulations and SCADA measurements is important. This comparison between SCADA-measured power and the 10-minute mean of simulated power for UW and UU is depicted in Figure 4.5a.



(a) Comparison of SCADA-measured power and simulated power generation



(b) Range (min to max) of generated power for both SCADA measurements and simulations

Figure 4.5: Comparison of SCADA-measured and simulated power generation for UW and UU joint distributions

In Figure 4.5a, the x axis indicates the mean wind speeds of SCADA measurement and simulations. Each data point on the graph is a 10-minute mean of generated power, either from SCADA measurements or simulations. Since each simulation duration is 10 minutes, the displayed value is the mean generated power for that interval. The solid lines on the graph represent the power averaged for each wind speed bin, as explained in Section 4.2.1. The MinMax scaling is based on the binned average power

for both SCADA and simulation. There are notable distinctions between Figure 4.5a and Figure 4.5b in terms of their data representation and scaling methodologies. Figure 4.5a specifically utilizes 10-minute *average* values from both the SCADA system and simulations. These values are normalized using the *MinMax* scaling approach. In contrast, Figure 4.5b focuses on the 10-minute *min* and *max* values obtained from the SCADA system and simulations. The scaling applied here ensures that the *mean value of generated power* is standardized to one. This distinction in data selection and scaling techniques underscores the different analytical perspectives offered by each figure. It is worth mentioning that values equal to zero in Figure 4.5a represent the minimum power generation and not zero generated power, as the min-max scaling output ranges between zero and one.

While the plots demonstrate a good match between simulation and measurements, the measured data exhibits more variability than the simulations, which is expected due to the variability in turbine behaviour within 10-minute intervals. We computed the difference between the scaled binned averaged measured and simulated power to assess their disparity. The results for a selection of wind speeds are shown in Table 4.4. The difference in the UU case for $5m/s$ is higher than it appears on the plot, as the scaled power values for measurements and simulations are both in the range of 0.01 to 0.02.

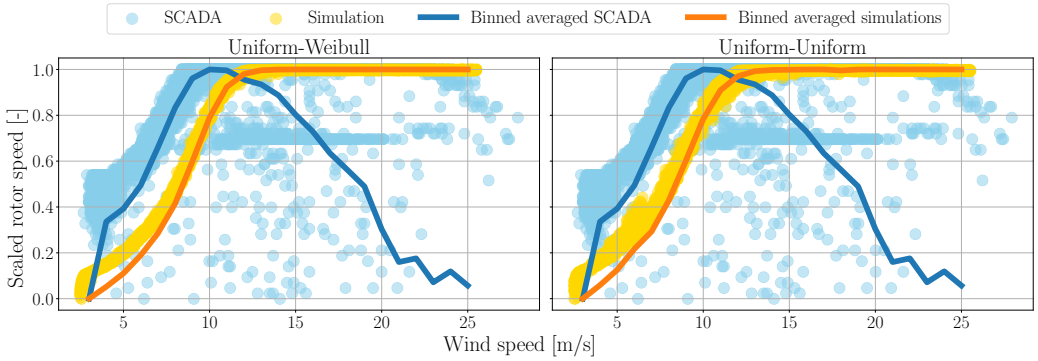
Table 4.4: Difference between binned scaled simulated power and measured power in percentage across a selection of wind speeds for UW and UU joint distributions

Wind Speed [m/s]	Uniform-Weibull[%]	Uniform-Uniform[%]
3	0	0
5	8.63	26.86
10	-11.99	-10.97
15	2.65	2.17
20	6.20	6.20
25	25.03	25.02

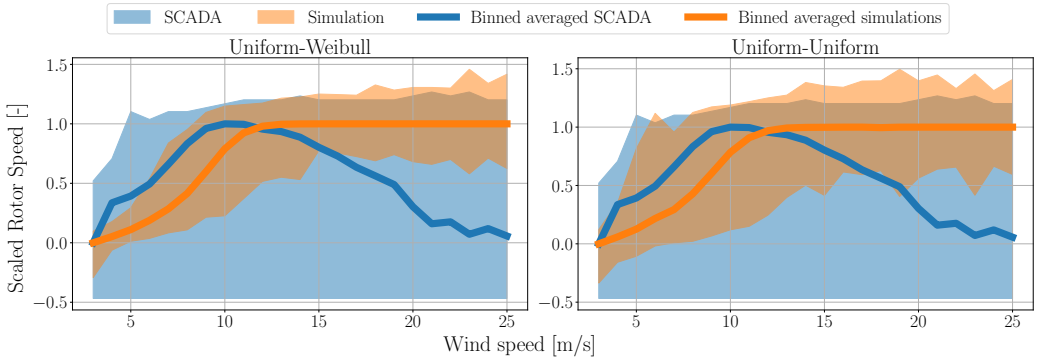
Figure 4.5b displays the range between the minimum and maximum power from SCADA measurements and simulations. The upper and lower bounds are based on the measurements or simulations' 10-minute maximum and minimum values. The shaded area indicates the range for the simulations and SCADA data. As shown in

Figure 4.5b, the range is wider for SCADA measurements than for simulations, which is anticipated.

Another indicator for assessing the alignment of the utilized model with real-world turbine behaviour is the rotor speed. Similarly, Figures 4.6a and 4.6b compare the 10-minute mean and range for the rotor speed. These figures show a notable discrepancy between simulations and SCADA measurements. There are two reasons for this observation: a) for below-rated wind speeds, the controller behaviour of the model and the SCADA-measured turbine differs; b) for above-rated wind speeds, we lack sufficient measurement data, and the available data are based on unknown parameters that reduce the mean rotor speed. The limited availability of SCADA data and lack of variables such as pitch angle restrict a more accurate understanding of wind turbine behaviour. Although there is a discrepancy between the model and SCADA rotor speed, the effect on the purpose is insignificant, as the rotor speed is mainly a function of wind speed and would not affect the SCADA load measurement from the turbine. It is important to note that the aim of this work is not to produce a fully accurate wind turbine simulator, but rather a practical relative health assessment tool when the manufacturer's turbine model is unavailable.



(a) Comparison of SCADA-measured rotor speed and simulated rotor speed



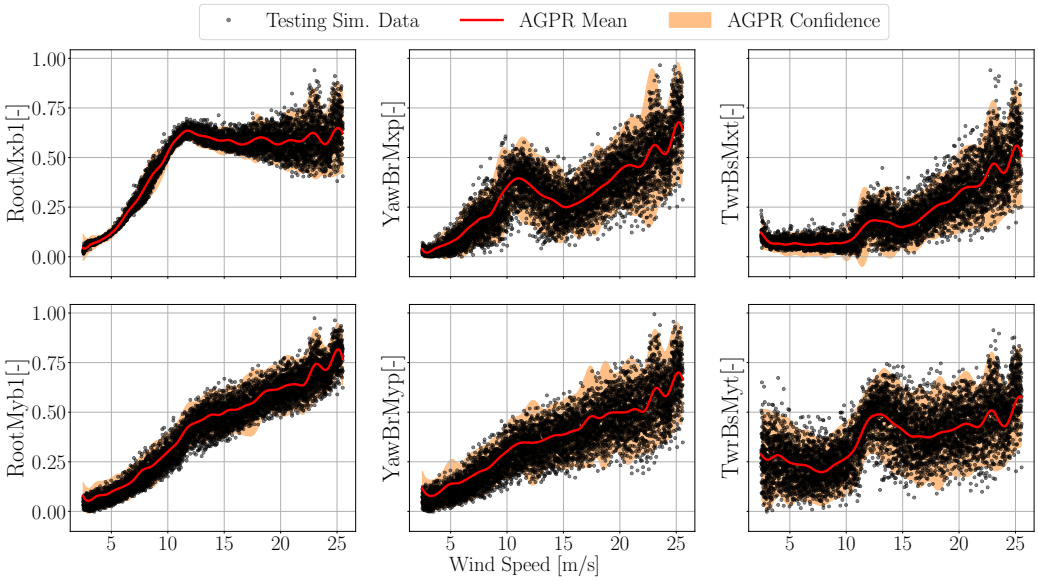
(b) Range (min to max) of rotor speed for both SCADA measurements and simulations

Figure 4.6: Comparison of SCADA-measured and simulated rotor speed for UW and UU joint distributions

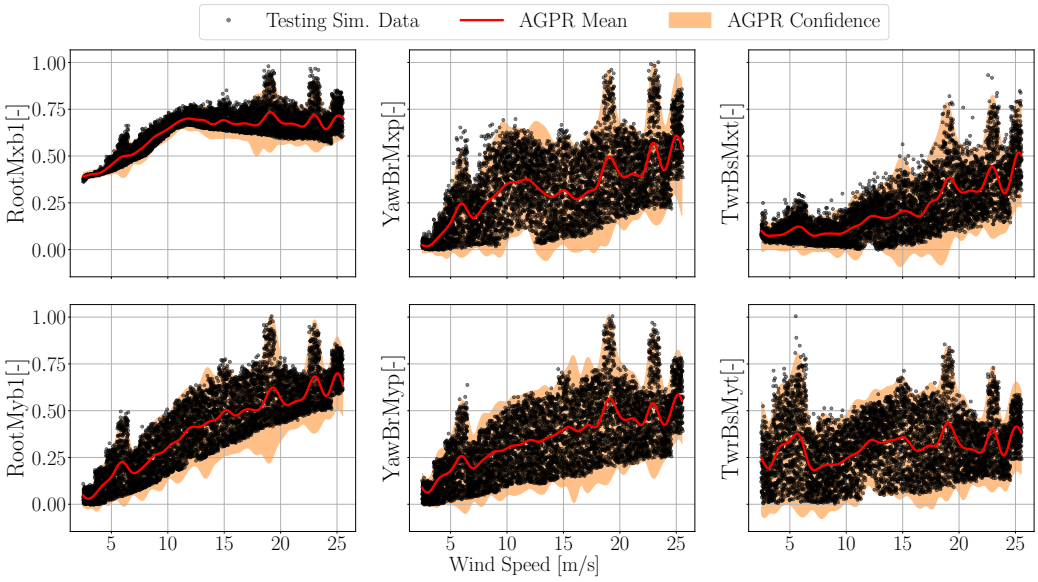
4.3.6 AGPR testing results - Hybrid simulations

After training the AGPR models, we evaluated them by supplying a high-resolution array of wind speeds ranging from the cut-in to cut-out wind speeds. The trained models then rapidly provided the mean and STD of the DEL for each wind speed. To assess the accuracy of these models, we employed two approaches. Firstly, we plotted the evaluation process's mean and confidence interval output alongside all the simulation testing dataset to visually assess whether the testing data fell within the confidence intervals. Secondly, we binned the simulation testing data and calculated the mean and STD of the testing QoI, as described in Section 4.2.1. We then determined the difference between the testing data and AGPR output using KL divergence. Figures 4.7a and 4.7b display the AGPR output's mean, confidence interval, and the testing data for the DEL. In the following figures through this manuscript,

the confidence interval is defined ± one STD from the AGPR mean.



(a) UW joint distribution



(b) UU joint distribution testing

Figure 4.7: The AGPR outputs and simulation testing data for UW and UU joint distributions

Visual inspection of the plots in Figure 4.7 reveals that the AGPR accurately predicts the mean and confidence interval. Notably, there is a variation in turbine

behaviour in terms of DEL between the UW and UU cases. The UW DEL exhibits a broader output range and is less symmetrical around the mean value. Owing to this asymmetry and the symmetric nature of the confidence interval, the lower bound of the confidence interval in the UU case overshoots the testing data. The broader range of behaviour in the UU case is expected, as the sampling of mean wind speed and STD fully covers the domain between the extreme minimum and maximum values per bin. Figure 4.7 shows some peaks in the confidence interval, particularly for the UU case, indicating a successful implementation of PLL. As explained in Section 4.2.5, the PLL estimates the posterior distribution of the output data. In our case, the output data has a cluster of peaks, and the posterior distribution is expanded in terms of STD to cover those wind speeds. This expansion ensures that the simulation training and testing data is adequately covered.

Table 4.5 presents the KL divergence between the AGPR output and the testing data for three wind speeds, with the testing data treated as the ground truth. The KL divergence is calculated according to Equation (4.6).

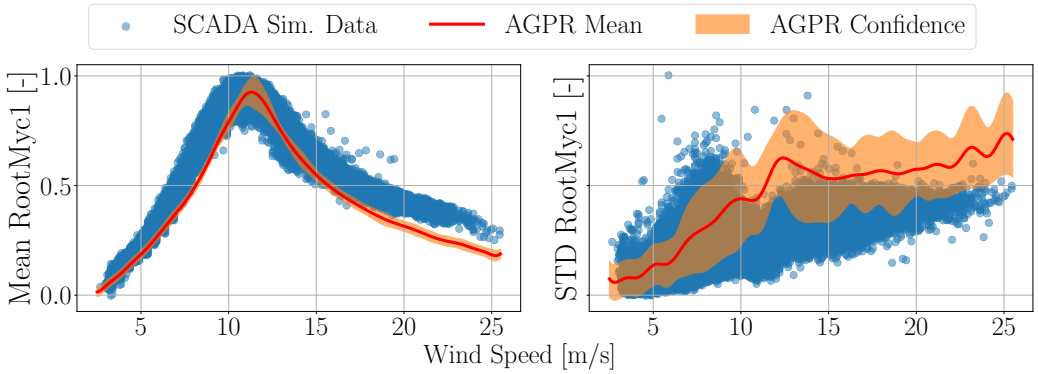
Table 4.5: KL divergence for three wind speeds comparing testing data and AGPR output

	UW [-]			UU [-]		
Channels	6 m/s	12 m/s	18 m/s	6 m/s	12 m/s	18 m/s
RootMxb1	0.029	0.000	0.024	0.002	0.004	0.042
RootMyb1	0.010	0.009	0.014	0.003	0.001	0.005
YawBrMxp	0.043	0.013	0.001	0.002	0.005	0.024
YawBrMyp	0.004	0.007	0.020	0.003	0.001	0.013
TwrBsMxt	0.009	0.025	0.005	0.001	0.000	0.003
TwrBsMyt	0.002	0.000	0.011	0.005	0.004	0.001

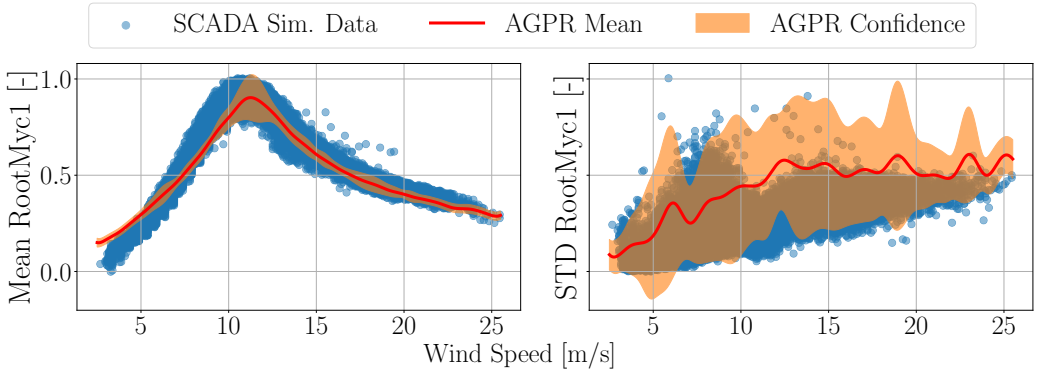
The KL divergence values in Table 4.5 indicate a good match between the AGPR output and the testing data. However, the AGPR is less accurate in the UU case, due to the reasons previously mentioned. Moreover, the KL divergence varies depending on the output channel and wind speed. A comparison of the plots in Figure 4.7 and Table 4.5 suggests this is primarily due to the overestimation of the STD by the AGPR. These results are typical of other wind speed bins.

4.3.7 AGPR testing results - SCADA measurement

In Table 4.2, we provided one moment and three accelerations, each processed with a 10-minute mean. These outputs correspond to the moment and acceleration available in our SCADA database, as shown in Table 4.1. We trained a AGPR for the 10-minute mean and STD of each output based on hybrid simulations output. Subsequently, we tested the accuracy of the trained models against the corresponding mean and STD of the SCADA measurement data fields. Let us reiterate that the primary objective is to assess the effectiveness of normalized simulations of a specific simulated wind turbine in predicting the behaviour of a real wind turbine machine with its own blade, controller, and structure when the real-world turbine data is proprietary and unavailable. The AGPR training settings are mentioned in Table 4.3. Figure 4.8 presents the AGPR output against the SCADA measurement for the OFP BR moment.



(a) UW joint distribution



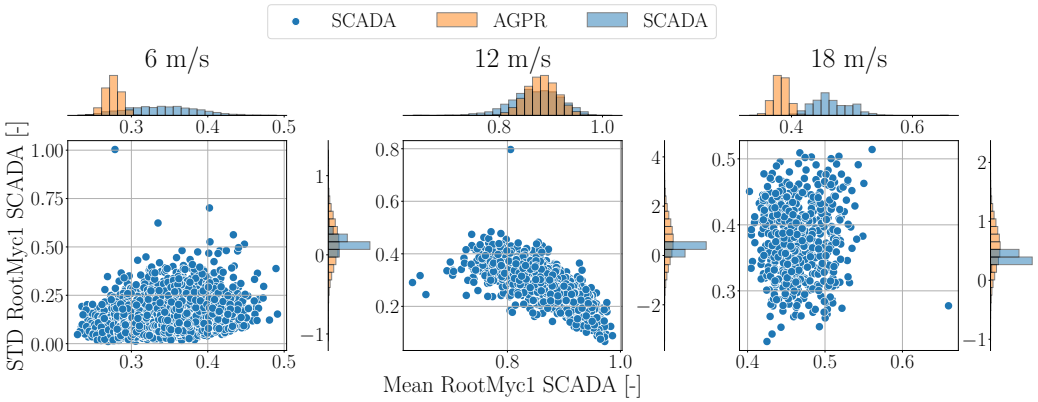
(b) UU joint distribution

Figure 4.8: OFP BR AGPR prediction vs the SCADA data

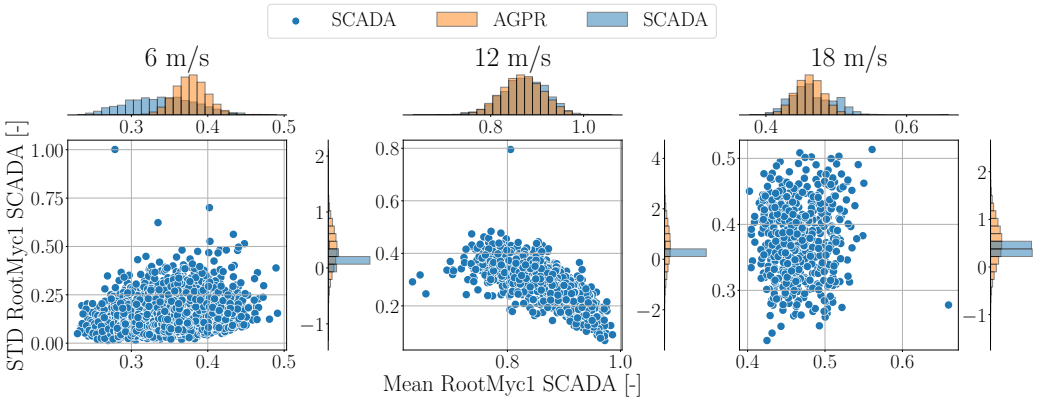
Despite the discrepancies in controller behaviour mentioned in Section 4.3.5, the mean and STD of the moments from the AGPR and SCADA follow a similar pattern overall. The mean value plots in Figure 4.8 show that the AGPR output's confidence interval does not entirely cover the SCADA data. This discrepancy might be attributed to the blade in the model being stiffer than in reality. This hypothesis is also supported by the STD plots, where the confidence interval of UU joint distribution provides better coverage over the SCADA data. The UU joint distribution's broader range, which encompasses unrealistic combinations of mean wind speed and STD, results in a closer approximation of reality by the AGPR prediction. This necessity for excessive turbulence intensity is likely due to the stiffer modelled blade structure.

A noticeable peak in the mean trajectory for the STD around $6m/s$ in Figure 4.8b is not observed in Figure 4.8a. A similar peak is also visible in Figure 4.7b for all the outputs except RootMxb1. This peak arises because the UU case models the turbine with a broader sample of STD at wind speeds around $6m/s$. According to the Campbell diagram of the NREL 5MW turbine [43], the blade passing frequency matches the side-side and fore-aft frequency of the tower at this wind speed. Typically, the STD of the wind speed does not often enter the resonance region in 10-minute simulations. However, with higher turbulence, the turbine enters and exits the resonance region frequently, increasing the mean, STD, and DEL of the simulation.

To provide a more concise comparison, Figure 4.9 compares the SCADA data with the AGPR predictions for three wind speeds. For each wind speed, we binned the SCADA data as explained in Section 4.2.1, using the AGPR output at the bin center for comparison.



(a) UW joint distribution



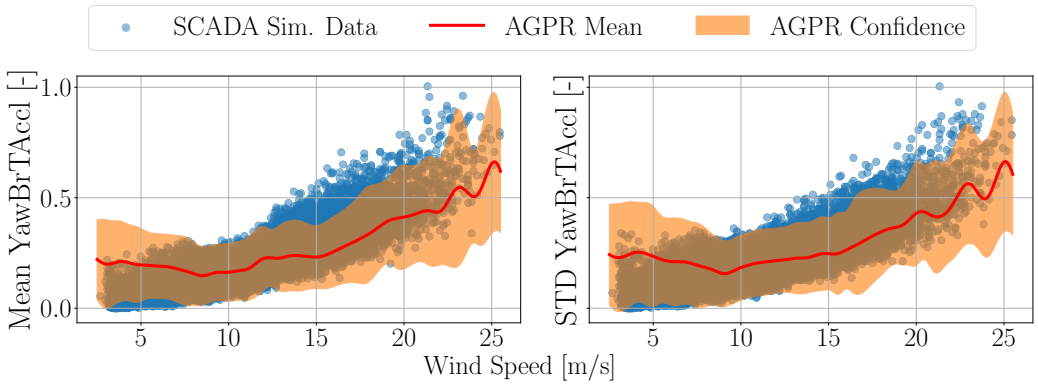
(b) UU joint distribution

Figure 4.9: OFP BR AGPR prediction vs the SCADA for three wind speeds. The histogram represents the probability density.

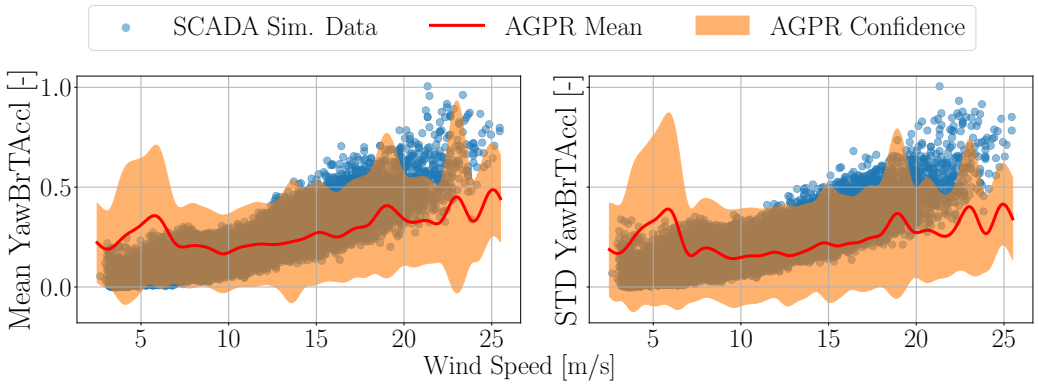
The histograms for the AGPR in Figure 4.9 are based on 10^5 samples from the normal distribution provided by the AGPR at the bin center. Figure 4.9 demonstrates that the AGPR fit varies as a function of both the wind speed and the joint distribution. The AGPR prediction of the mean value is more accurate for the UU case, but the STD prediction histogram is significantly broader than the SCADA data. This broader range is due to the manner of presenting the results: the confidence interval in Figure 4.8 is twice the STD, while in Figure 4.9, the histogram covers a wider range. It may not be a fair comparison to take the AGPR output at the center of the bin and compare it with the SCADA, which has values all over the bin. However, there are two reasons behind this. Firstly, the width of the bin for $1m/s$ is narrow, and looking at Figure 4.8, it shows that the AGPR result and SCADA data do not change significantly in the span of $1m/s$. Therefore, one AGPR model would suffice.

Secondly, as we recall from Figure 4.3, the measured wind speed is clustered at bins. Hence, using SCADA for one wind speed value limits the number of available data, making the statistical comparison inaccurate.

The other SCADA data fields that match the OpenFAST output is the TT resultant acceleration. Figure 4.10 presents the AGPR output for TT resultant acceleration trained on hybrid simulation output against the SCADA measurements.



(a) UW joint distribution



(b) UU joint distribution

Figure 4.10: TT resultant acceleration AGPR prediction vs the SCADA data.

In Figure 4.10, the same peak around $6m/s$ observed in the mean trajectory in Figure 4.10b is attributable to the broader sampling of STD at wind speeds around $6m/s$ as explained in Section 4.3.6. It is worth mentioning that the confidence interval, represented by the mean value $\pm$STD, does not encompass the entire spread of the AGPR output. In other words, these intervals do not serve as a measure of

uncertainty quantification. Rather, their primary function is to delineate the area where 68.3% of the AGPR outputs are located. This is more visible in Figure 4.9 when the AGPR output spread is wider than the SCADA measurement.

Figure 4.10 illustrates that the UU case provides broader coverage in terms of the confidence interval due to the broader STD samples. However, the UW case more accurately follows the acceleration trend for the resultant acceleration. Assuming similar rotor thrust forces, the TT acceleration is mainly a function of the structural stiffness of the tower. A closer approximation to reality for the UW case, which is deemed more realistic for wind speed STD, suggests that the modelled tower structural stiffness is probably closer to the turbine for which we had access to the SCADA data. A more precise verification could have been achieved if we had access to tower frequency readings in the SCADA data.

We calculated the KL divergence for the OFP blade root moments and the tower top acceleration to facilitate a more precise comparison, as presented in Table 4.6. This calculation, based on Equation (4.7), uses M as the binned SCADA data described in Section 4.2.1, and G is derived from taking 10^5 samples from the normal distribution provided by AGPR at the center of the bin. Although Table 4.6 could be extended to include all wind speeds for which we have SCADA data, we have limited the presentation to three wind speeds for brevity.

Table 4.6: KL divergence for three wind speeds comparing binned SCADA and AGPR center bin output

	UW [-]			UU [-]		
Data field	6 m/s	12 m/s	18 m/s	6 m/s	12 m/s	18 m/s
OFP BR Moment mean	1.783	0.160	9.317	0.701	0.032	0.184
OFP BR Moment STD	3.006	5.556	5.736	3.683	6.196	5.487
TT Res. accl. mean	1.680	0.253	0.915	5.358	0.889	1.539
TT Res. accl. STD	2.363	0.252	0.625	5.511	1.619	2.884

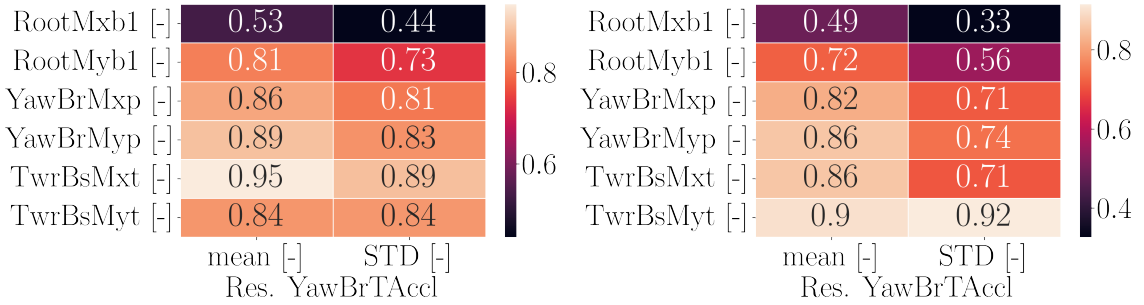
Considering KL divergence as an error metric, the most accurate AGPR predictions occur for the resultant tower top accelerations in the UW case for both mean and STD. This indicates that the output of the AGPR depends on the joint distribution selected for sampling from the specific data field under examination. It also suggests

that the AGPR model trained on either UW or UU may perform better depending on the data field, which is influenced by the wind turbine's controller, aerodynamics, and structural behaviour. We lack access to the model from the manufacturer, so we cannot precisely identify the reasons for the variety in AGPR performance. Nevertheless, this model and method demonstrate promising results as a primary health assessment tool.

4.3.8 How can we use this model?

To this point, we have trained a series of AGPR models on hybrid simulations for DEL, mean, and STD. We have shown that the AGPR models trained on these hybrid simulations can accurately predict the DEL. Furthermore, we have demonstrated that the AGPR models can predict the OFP BR moments and TT accelerations from SCADA data with acceptable identification of trends. The next step is to understand how these models can be interconnected.

Figure 4.11 provides correlation heat maps for both UW and UU joint distributions, illustrating the correlation between DEL and the TT resultant acceleration mean.



(a) UW joint distribution

(b) UU joint distribution

Figure 4.11: Hybrid simulations correlation heatmap. The rows represent the DEL values.

Figure 4.11 shows a strong correlation between the resultant TT acceleration and the DEL loads along the tower. Except for the blade root edge-wise moment, other channels are highly or very highly correlated with the resultant accelerations. Given

the high correlation between the resultant TT acceleration and the DEL, and the fidelity of the AGPR in predicting the SCADA resultant TT (Figure 4.10), we can hypothesize that AGPR predictions of DEL would correlate with actual DEL measurements. Although this medium level of accuracy in DEL prediction based on AGPR is insufficient for a precise health assessment, it can serve as an indicator and primary tool for diagnosing the health of wind turbine assets. During this study, we did not have access to moment measurement time series along the tower to validate this hypothesis. The authors acknowledge this limitation and recognize the need to test the hypothesis with a wind turbine instrumented with an array of sensors.

4.4 Conclusions

This work emphasizes the crucial need for assessing the structural health of wind turbines, given their operation in harsh environmental conditions. This assessment is vital for ensuring the longevity and optimal performance of wind turbines. The focus is on understanding the fatigue damage accumulation in these structures and the importance of advanced methods for accurate prediction and monitoring of their structural health. The methodology section details the construction of a AGPR model using a combination of the SCADA wind measurement and simulations utilizing Sobol's sampling method. It presents a systematic method involving the generation of hybrid simulation, followed by its use in modelling and validation. The final section first explains the conditions under which the results were generated, followed by a presentation and discussion of these findings. Then, it shows the accuracy of the utilized model by comparing the simulation output and SCADA measurement for the generated power and rotor speed. Afterwards, the section shows how a AGPR trained on the hybrid simulations database accurately predicts the testing data set and the SCADA acceleration and loads data fields. It also explores the model's accuracy, reliability, and ability to predict potential structural issues. The results are not just numerical outputs; they are interpreted to provide insights into the overall effectiveness of the AGPR model in real-world scenarios.

This research conclusively demonstrates the efficacy and robustness of AGPR in the realm of wind turbine asset reliability. The research underscores the advanced predictive capabilities of AGPR, particularly in handling the heteroscedasticity in-

herent in wind turbine operational data. The ability of AGPR to accurately model and predict loads and accelerations based on a range of inputs, especially those derived from SCADA systems, while it is trained on the publicly available models and methodology is a noticeable development. It is worth noting that our tests have revealed that the choice of publicly available wind turbine model impacts the accuracy and compatibility of the trained AGPR model with the SCADA measurements. This study elaborates on how AGPR is effectively trained on hybrid simulation data sets, blending real-world measurements with simulated data to create a comprehensive model. This approach allows a better understanding of the data and enhances the model's predictive accuracy. The research highlights the validation processes the AGPR model underwent, affirming its reliability and accuracy. The performance of the AGPR models in predicting the loads under various operational scenarios showcases its practical applicability in real-world settings. The paper points out the potential for AGPR to serve as a standard tool in the predictive maintenance of wind turbines.

While the AGPR model demonstrates promising capabilities in predicting DEL, it is important to recognize the challenges faced in this study. The lack of data about the measurement of turbine natural frequencies, dependence on specific data fields, and the limitations posed by the unavailability of the measurement wind turbine model are notable challenges. Furthermore, the transferability of the model and whether the model can be applied to various turbine types and operational scenarios, including extreme events, remains to be tested.

4.4.1 Future work

Considering the fact that the model employed in this study is publicly available, and the turbine manufacturer's model is not accessible, the performance of the AGPR for the purposes of this work is promising. For future research, there is an ambition to extend this study and test the hypothesis mentioned in Section 4.3.8 on a turbine equipped with sensors. Additionally, having access to more extensive and varied data fields in SCADA would be beneficial. This would help in reducing measurement uncertainties, particularly at higher wind speeds, and in fine-tuning the model for outputs that more closely reflect reality. Ideally, access to the actual wind turbine model would substantially enrich this research. Also, it is important to demonstrate

the generalizability of this approach by implementing it on an offshore wind turbine in future work.

In the realm of data-driven modelling, we employed AGPR for this study with remarkable results. Nevertheless, exploring other methods, such as probabilistic neural networks or Bayesian neural networks, is crucial. Another avenue could involve moving away from probabilistic models and experimenting with ANNs to map environmental inputs to loads. Additionally, considering SCADA data as a time series and employing a transformer, as discussed in [60], could provide a novel approach to building a data-driven model that predicts loads based on a limited series of environmental and controller inputs.

In this study, our focus was primarily on data from power production DLCs. However, other significant events, such as shutdowns, gusts, and faults during a wind turbine's lifetime, can impact its structural integrity. These scenarios are crucial and should be considered in future studies.

Funding

We greatly acknowledge the funding for this study by the Natural Sciences and Engineering Research Council of Canada (NSERC) and MiTACS.

Data availability

The SCADA data and hybrid databases are unavailable due to NDA. The developed code for GPR is available upon request.

Acknowledgments

This research was partly enabled by support provided by the Digital Research Alliance of Canada (alliancecan.ca). We want to thank John Wang and Selena Farris from Clir for providing us with SCADA data sets and giving us feedback during this study.

Conflicts of interest

The authors declare no conflict of interest.

Bibliography

[1] Chia Chen Ciang, Jung-Ryul Lee, and Hyung-Joon Bang. Structural health monitoring for a wind turbine system: A review of damage detection methods. *Measurement Science and Technology*, 19(12):122001, 2008. ISSN 0957-0233. doi: 10.1088/0957-0233/19/12/122001.

[2] Maria Martinez-Luengo, Athanasios Kolios, and Lin Wang. Structural health monitoring of offshore wind turbines: A review through the Statistical Pattern Recognition Paradigm. *Renewable and Sustainable Energy Reviews*, 64:91–105, 2016. ISSN 1364-0321. doi: 10.1016/j.rser.2016.05.085.

[3] Wenxian Yang, Peter J. Tavner, Christopher J. Crabtree, Y. Feng, and Y. Qiu. Wind turbine condition monitoring: Technical and commercial challenges. *Wind Energy*, 17(5):673–693, 2014. ISSN 1099-1824. doi: 10.1002/we.1508.

[4] Pierre Tchakoua, René Wamkeue, Mohand Ouhrouche, Fouad Slaoui-Hasnaoui, Tommy Andy Tameghe, and Gabriel Ekemb. Wind Turbine Condition Monitoring: State-of-the-Art Review, New Trends, and Future Challenges. *Energies*, 7 (4):2595–2630, 2014. ISSN 1996-1073. doi: 10.3390/en7042595.

[5] Kevin Leahy, Colm Gallagher, Peter O'Donovan, and Dominic T. J. O'Sullivan. Issues with Data Quality for Wind Turbine Condition Monitoring and Reliability Analyses. *Energies*, 12(2):201, 2019. ISSN 1996-1073. doi: 10.3390/en12020201.

[6] B. Badrzadeh, M. Bradt, N. Castillo, R. Janakiraman, R. Kennedy, S. Klein, T. Smith, and L. Vargas. Wind power plant SCADA and controls. In *2011 IEEE Power and Energy Society General Meeting*, pages 1–7. doi: 10.1109/PES.2011.6039418.

[7] Pere Marti-Puig, Alejandro Blanco-M., Moisès Serra-Serra, and Jordi Solé-

Casals. Wind Turbine Prognosis Models Based on SCADA Data and Extreme Learning Machines. 11(2):590. ISSN 2076-3417. doi: 10.3390/app11020590.

[8] Jannis Tautz-Weinert and Simon J. Watson. Using SCADA data for wind turbine condition monitoring – a review. *IET Renewable Power Generation*, 11(4):382–394, 2017. ISSN 1752-1424. doi: 10.1049/iet-rpg.2016.0248.

[9] Henrik Stensgaard Toft, Lasse Svenningsen, Wolfgang Moser, John Dalsgaard Sørensen, and Morten Lybech Thøgersen. Assessment of wind turbine structural integrity using response surface methodology. *Engineering Structures*, 106:471–483, 2016. ISSN 0141-0296. doi: 10.1016/j.engstruct.2015.10.043.

[10] Gordon Stewart. Design Load Analysis of Two Floating Offshore Wind Turbine Concepts. *Doctoral Dissertations*, 2016. doi: 10.7275/7627466.0.

[11] Rui Teixeira, Alan O'Connor, Maria Nogal, Nandakumar Krishnan, and James Nichols. Analysis of the design of experiments of offshore wind turbine fatigue reliability design with Kriging surfaces. *Procedia Structural Integrity*, 5:951–958, 2017. ISSN 2452-3216. doi: 10.1016/j.prostr.2017.07.132.

[12] Kolja Müller, Martin Dazer, and Po Wen Cheng. Damage Assessment of Floating Offshore Wind Turbines Using Response Surface Modeling. *Energy Procedia*, 137: 119–133, 2017. ISSN 1876-6102. doi: 10.1016/j.egypro.2017.10.339.

[13] Kolja Müller and Po Wen Cheng. Application of a Monte Carlo procedure for probabilistic fatigue design of floating offshore wind turbines. *Wind Energy Science*, 3(1):149–162, 2018. ISSN 2366-7443. doi: 10.5194/wes-3-149-2018.

[14] Nikolay Dimitrov, Mark C. Kelly, Andrea Vignaroli, and Jacob Berg. From wind to loads: Wind turbine site-specific load estimation with surrogate models trained on high-fidelity load databases. *Wind Energy Science*, 3(2):767–790, 2018. ISSN 2366-7443. doi: 10.5194/wes-3-767-2018.

[15] Rad Haghi and Curran Crawford. Surrogate models for the blade element momentum aerodynamic model using non-intrusive polynomial chaos expansions. *Wind Energy Science*, 7(3):1289–1304, 2022. ISSN 2366-7443. doi: 10.5194/wes-7-1289-2022.

[16] Laura Schröder, Nikolay Krasimirov Dimitrov, and David Robert Verelst. A surrogate model approach for associating wind farm load variations with turbine failures. *Wind Energy Science*, 5(3):1007–1022, 2020. ISSN 2366-7443. doi: 10.5194/wes-5-1007-2020.

[17] Nikolay Dimitrov and Tuhfe Göçmen. Virtual sensors for wind turbines with machine learning-based time series models. *Wind Energy*, 25(9):1626–1645, 2022. ISSN 1099-1824. doi: 10.1002/we.2762.

[18] Jorge Maldonado-Correa, Sergio Martín-Martínez, Estefanía Artigao, and Emilio Gómez-Lázaro. Using SCADA Data for Wind Turbine Condition Monitoring: A Systematic Literature Review. *Energies*, 13(12):3132, 2020. ISSN 1996-1073. doi: 10.3390/en13123132.

[19] Elena Gonzalez, Bruce Stephen, David Infield, and Julio J. Melero. Using high-frequency SCADA data for wind turbine performance monitoring: A sensitivity study. *Renewable Energy*, 131:841–853, 2019. ISSN 0960-1481. doi: 10.1016/j.renene.2018.07.068.

[20] Christopher S. Gray and Simon J. Watson. Physics of Failure approach to wind turbine condition based maintenance. *Wind Energy*, 13(5):395–405, 2010. ISSN 1099-1824. doi: 10.1002/we.360.

[21] Christos Galinos, Nikolay Dimitrov, Torben J. Larsen, Anand Natarajan, and Kurt S. Hansen. Mapping Wind Farm Loads and Power Production - A Case Study on Horns Rev 1. *Journal of Physics: Conference Series*, 753(3):032010, 2016. ISSN 1742-6596. doi: 10.1088/1742-6596/753/3/032010.

[22] Eduardo J. Alvarez and Adrijan P. Ribaric. An improved-accuracy method for fatigue load analysis of wind turbine gearbox based on SCADA. *Renewable Energy*, 115:391–399, 2018. ISSN 0960-1481. doi: 10.1016/j.renene.2017.08.040.

[23] W. Dheelibun Remigius and Anand Natarajan. Identification of wind turbine main-shaft torsional loads from high-frequency SCADA (supervisory control and data acquisition) measurements using an inverse-problem approach. *Wind Energy Science*, 6(6):1401–1412, 2021. ISSN 2366-7443. doi: 10.5194/wes-6-1401-2021.

[24] Ravi Pandit, Davide Astolfi, Jiarong Hong, David Infield, and Matilde Santos. SCADA data for wind turbine data-driven condition/performance monitoring: A review on state-of-art, challenges and future trends. *Wind Engineering*, 47(2): 422–441, 2023. ISSN 0309-524X. doi: 10.1177/0309524X221124031.

[25] Luis Vera-Tudela and Martin Kühn. Analysing wind turbine fatigue load prediction: The impact of wind farm flow conditions. *Renewable Energy*, 107:352–360, 2017. ISSN 0960-1481. doi: 10.1016/j.renene.2017.01.065.

[26] Anand Natarajan and Leonardo Bergami. Determination of wind farm life consumption in complex terrain using ten-minute SCADA measurements. *Journal of Physics: Conference Series*, 1618(2):022013, 2020. ISSN 1742-6596. doi: 10.1088/1742-6596/1618/2/022013.

[27] Charilaos Mylonas, Imad Abdallah, and Eleni Chatzi. Conditional variational autoencoders for probabilistic wind turbine blade fatigue estimation using Supervisory, Control, and Data Acquisition data. *Wind Energy*, 24(10):1122–1139, 2021. ISSN 1099-1824. doi: 10.1002/we.2621.

[28] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005. ISBN 978-0-262-25683-4. doi: 10.7551/mitpress/3206.001.0001.

[29] Ravi Kumar Pandit and David Infield. SCADA-based wind turbine anomaly detection using Gaussian process models for wind turbine condition monitoring purposes. *IET Renewable Power Generation*, 12(11):1249–1255, 2018. ISSN 1752-1424. doi: 10.1049/iet-rpg.2018.0156.

[30] Yanting Li, Shujun Liu, and Lianjie Shu. Wind turbine fault diagnosis based on Gaussian process classifiers applied to operational data. *Renewable Energy*, 134: 357–366, 2019. ISSN 0960-1481. doi: 10.1016/j.renene.2018.10.088.

[31] Jürgen Herp, Mohammad H. Ramezani, Martin Bach-Andersen, Niels L. Pedersen, and Esmaeil S. Nadimi. Bayesian state prediction of wind turbine bearing failure. *Renewable Energy*, 116:164–172, 2018. ISSN 0960-1481. doi: 10.1016/j.renene.2017.02.069.

[32] Luis David Avendaño-Valencia, Imad Abdallah, and Eleni Chatzi. Virtual fatigue diagnostics of wake-affected wind turbine via Gaussian Process Regression.

Renewable Energy, 170:539–561, 2021. ISSN 0960-1481. doi: 10.1016/j.renene.2021.02.003.

[33] David Wilkie and Carmine Galasso. Gaussian process regression for fatigue reliability analysis of offshore wind turbines. *Structural Safety*, 88:102020, 2021. ISSN 0167-4730. doi: 10.1016/j.strusafe.2020.102020.

[34] D. Singh, R. P. Dwight, K. Laugesen, L. Beaudet, and A. Viré. Probabilistic surrogate modeling of offshore wind-turbine loads with chained Gaussian processes. *Journal of Physics: Conference Series*, 2265(3):032070, 2022. ISSN 1742-6596. doi: 10.1088/1742-6596/2265/3/032070.

[35] Max Kuhn and Kjell Johnson. *Feature Engineering and Selection: A Practical Approach for Predictive Models*. URL `http://www.feat.engineering/`.

[36] Roger Daley. *Atmospheric data analysis*. Number 2. Cambridge university press, 1993.

[37] I. M Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967. ISSN 0041-5553. doi: 10.1016/0041-5553(67)90144-9.

[38] Sergei Kucherenko, Daniel Albrecht, and Andrea Saltelli. Exploring multi-dimensional spaces: A comparison of latin hypercube and quasi monte carlo sampling techniques. 2015.

[39] Marissa Renardy, Louis R. Joslyn, Jess A. Millar, and Denise E. Kirschner. To Sobol or not to Sobol? The effects of sampling schemes in systems biology applications. *Mathematical biosciences*, 337:108593, 2021. ISSN 0025-5564. doi: 10.1016/j.mbs.2021.108593.

[40] Bonnie J Jonkman. TurbSim user's guide: Version 1.50. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2009.

[41] Bonnie Jonkman, Rafael M Mudafort, Andy Platt, E. Branlard, Mike Sprague, jjonkman, HaymanConsulting, Matt Hall, Ganesh Vijayakumar, Marshall Buhl, Hannah Ross, Pietro Bortolotti, marco, Shreyas Ananthan, Michael S., Jon Rood, rdamiani, nrmendoza, sinolonghai, pschuenemann, Derek Slaughter, ashesh2512, kshaler, Stein Housner, psakievich, Kurt Bendl, Lucas Carmo, Eliot

Quon, mattrphillips, and Nobuhiro KUSUNO. Openfast/openfast: Openfast v3.3.0.

[42] IEC 61400-1. Wind energy generation systems - Part 1: Design requirements. Standard, International Electrotechnical Commission, 2019.

[43] Jason Jonkman, Sandy Butterfield, Walter Musial, and George Scott. Definition of a 5-MW reference wind turbine for offshore system development. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2009.

[44] Jennifer Rinker and Katherine Dykes. WindPACT Reference Wind Turbines. Technical Report NREL/TP–5000-67667, 1432194.

[45] Pietro Bortolotti, Helena Canet Tarres, Katherine Dykes, Karl Merz, Latha Sethuraman, David Verelst, and Frederik Zahle. Iea wind task 37 on systems engineering in wind energy – wp2.1 reference wind turbines. Technical report, International Energy Agency, 2019. URL `https://www.nrel.gov/docs/fy19osti/73492.pdf`.

[46] Eliot Quon. NREL/openfast-turbine-models: A repository of OpenFAST turbine models developed by NREL researchers. `https://github.com/NREL/openfast-turbine-models/tree/master`, 2021. GitHub repository.

[47] Kenneth Thomsen. *The Statistical Variation of Wind Turbine Fatigue Loads*. Number 1063 in Risø-R. Risø National Laboratory, Roskilde, 1998. ISBN 978-87-550-2410-6 978-87-550-2411-3. URL `https://www.osti.gov/etdeweb/servlets/purl/292704`.

[48] Henrik Stiesdal. Rotor loadings on the BONUS 450 kW turbine. *Journal of Wind Engineering and Industrial Aerodynamics*, 39(1):303–315, 1992. ISSN 0167-6105. doi: 10.1016/0167-6105(92)90555-O.

[49] Masanori Matsuishi and Tatsuo Endo. Fatigue of metals subjected to varying stress. *Japan Society of Mechanical Engineers, Fukuoka, Japan*, 68(2):37–40, 1968.

[50] Joaquin Quiñonero-Candela, Carl Edward Rasmussen, and Christopher K. I. Williams. Approximation Methods for Gaussian Process Regression. In Léon

Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston, editors, *Large-Scale Kernel Machines*, pages 203–224. ISBN 978-0-262-25579-0. doi: 10.7551/mitpress/7496.003.0011.

[51] Kristian Kersting, Christian Plagemann, Patrick Pfaff, and Wolfram Burgard. Most likely heteroscedastic Gaussian process regression. In *Proceedings of the 24th International Conference on Machine Learning*, pages 393–400. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273546.

[52] Martin Jankowiak, Geoff Pleiss, and Jacob Gardner. Parametric Gaussian Process Regressors. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4702–4712. URL `https://proceedings.mlr.press/v119/jankowiak20a.html`.

[53] James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable Variational Gaussian Process Classification. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 351–360. URL `https://proceedings.mlr.press/v38/hensman15.html`.

[54] Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration. In *Advances in Neural Information Processing Systems*, volume 31. URL `https://proceedings.neurips.cc/paper/2018/hash/27e8e17134dd7083b050476733207ea1-Abstract.html`.

[55] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning Series. ISBN 978-0-262-01802-9.

[56] Art B. Owen. On dropping the first Sobol' point. URL `http://arxiv.org/abs/2008.08051`.

[57] Jason M Jonkman, Marshall L Buhl Jr, et al. FAST user's guide. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2005.

[58] Emmanuel Branlard. pyfast. `https://github.com/OpenFAST/python-toolbox`, 2023. GitHub repository.

[59] Jake Lever, Martin Krzywinski, and Naomi Altman. Model selection and overfitting. 13(9):703–704. ISSN 1548-7105. doi: 10.1038/nmeth.3968.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. URL `https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html`.

Chapter 5

Conclusions and future work

> *"If one day I look out from my cabin's porch and see a row of windmills spinning in the distance, I won't curse them. I will praise them. It will mean we are finally getting somewhere."*
>
> – David Suzuki

This chapter finalizes the dissertation by summarizing the main findings and the assumptions used in the research. It then discusses possible future work and concludes with *propositions* of this Ph.D. work.

5.1 Conclusion and main findings

Chapter 2 presented the development of a non-intrusive SM for time-marching aerodynamic simulations, utilizing PCE. The chapter detailed the aerodynamic model implemented in this study. Afterward, the methodology for constructing PCEs was outlined. A challenge in this work was overcoming the curse of dimensionality, which was addressed here by adopting a reduced Veers model. The study demonstrated that an increase in the number of simulations leads to statistically consistent results over time. This finding indicates that constructing a limited number of SMs, or potentially just one, is adequate for the length of the simulations. In essence, the passage of time has a minimal impact on sectional statistics. Therefore, it is feasible to build precise SMs by decreasing simulation lengths and increasing the number of simulations. The results showcased a fourth-order polynomial SM, based on 2002 simulations, each lasting 2 seconds, providing accurate results in extensive MCS. The efficiency of the

SM was due to its computational time effectiveness compared to aerodynamic simulations and the building of the model, alongside running MCS for precise statistics.

Chapter 3 explored using a sequential ML model to construct an SM that correlates high-resolution wind time series with the DEL of wind turbine components. The methodology involved two approaches: a) a three layer FCNN architecture that maps wind time series input variables to DEL as the reference case and b) a TCN-FCNN architecture for mapping synthetic wind time series to DEL. The process comprised defining input variables, generating synthetic wind time series, conducting aero-servo-elastic simulations, and calculating DEL. The initial phase involved defining the input variable space, followed by generating non-repetitive samples using QMC Sobol's sampling technique. Synthetic wind time series were then created using TurbSim, based on these samples. Subsequent aero-servo-elastic simulations were performed using an NREL 5MW reference wind turbine model and OpenFAST, adhering to IEC standards. The simulation output provided load time series data for various wind turbine components. DEL calculations were based on these data, using the Palmgren-Miner linear damage rule. The DEL database was then segmented into training and testing sets to train and evaluate the SMs. The results indicated that incorporating certain input features, such as tower top acceleration time series, improved the accuracy of predicting tower bottom side-side moments for TCN-FCNN architecture. The FCNN architecture, though simpler, offered competitive predictive accuracy. The study also assessed the TCN-FCNN model's capacity to predict DEL for turbines in wake conditions, a scenario unaddressed by the FCNN SM. Findings showed that TL on a limited dataset could accurately forecast the DEL of a turbine in the wake. A sensitivity analysis showed that both TCN-FCNN and FCNN SMs remained accurate with a small number of training samples, indicating their practical efficiency and versatility.

Chapter 4 exhibited the efficacy and robustness of AGPR in the realm of wind turbine asset reliability. The research highlighted the advanced predictive capabilities of AGPR, particularly in addressing the heteroscedasticity inherent in wind turbine operational and simulation data. The study elaborated on how AGPR is trained on hybrid datasets, which combine real-world measurements with simulated data to create a comprehensive model. This approach provides a reasonable understanding of the data and enhances the predictive accuracy of the AGPR model. AGPR shows the potential to accurately model and predict loads and accelerations based on a

range of inputs, including those obtained from SCADA systems, while being trained on publicly available model outputs. The validation of the AGPR model predictions research demonstrated its reliability and accuracy. The performance of the AGPR models in predicting loads under various operational scenarios highlights its practical applicability in real-world settings. The paper pointed out the potential for AGPR to serve as a primary tool in the predictive maintenance of wind turbines.

5.2 Research Assumptions

This research, similar to any scientific endeavour, is based on certain simplifying assumptions, which are integral to the modelling approaches in the various chapters. Future work will aim to address and potentially relax these assumptions. These are detailed as follows:

- In Chapter 2, I employed the NREL 5MW model, developed in Python. This model accurately represented the full NREL 5MW rotor but did not include a controller. For simplicity, the wake was modelled as static (frozen wake), as incorporating a dynamic wake was computationally intensive without significantly enhancing the research value.

- In Chapter 2, to circumvent the curse of dimensionality, a simplified version of the Veers model [1] was employed. Recognizing the limitations of this approach, in Chapters 3 and 4, I utilized TurbSim along with full-field turbulent wind.

- For Chapters 3 and 4, I used the complete NREL 5MW model with all degrees of freedom. Some simplifications were made in the simulations: yaw misalignment scenarios were not included, and only the power production DLC with the normal turbulence model was considered. These assumptions reduced the number of simulations required without notably impacting the SM's accuracy and performance.

- Chapter 3 introduces a simplified wake model. Given the complexity of wake dynamics, a critical topic in wind energy research, this model encapsulates the fundamental principles of wake theory while remaining straightforward to implement in simulations.

- In Chapter 4, a year's worth of SCADA data was analyzed, focusing exclusively on power production data where the generated power exceeded zero. This filtration was crucial for aligning simulations with SCADA measurements, eliminating irrelevant data points.
- Also, in Chapter 4, the AGPR length scale and the number of inducing points were kept constant across different output channel training sessions. While optimization of these parameters could potentially improve fit, preliminary studies indicated minimal gains relative to the effort required.
- The wind turbine model utilized in Chapter 4 was the unmodified NREL 5MW. This was primarily due to limited data availability regarding the specific turbine under study (e.g., tower frequency or controller settings).

5.3 Future work

Embarking on a PhD journey often evokes mixed feelings about future research directions. While there might be a tinge of regret for not exploring certain topics, there is also a sense of relief and accomplishment in passing the baton to future researchers.

- The overarching aim of this research has been to develop a comprehensive, versatile SM capable of mapping wind time series to load statistics or time series, applicable to any turbine type, under varying conditions and locations. Although achieving this goal extends beyond the scope of a single Ph.D. tenure, this dissertation has laid the groundwork for such an SM.
- I initially explored PCE as a SM, but due to its complexities and the curse of dimensionality, the work was limited. A major problem in PCE is the curse of dimensionality resulting from the number of random variables. In synthetic unsteady wind scenarios, particularly in wind turbine engineering, this number of random variables can reach the order of thousands [1]. A viable solution for identifying random variables in wind time series involves employing a variational autoencoder [8]. This approach allows for training a variational autoencoder to generate new sets of unsteady wind from distributions in its latent space, effectively reducing the need for thousands of random variables and inverse

Fourier transformations. The resulting probabilistic latent space could then be applied as input to PCE SMs.

- Revisiting PCE SMs for alternative applications is also essential. For instance, in wave energy engineering, the inherent nature of waves requires significantly fewer random variables, making PCE a suitable method for developing SMs for wave energy converters. Another potential application is airborne wind aerodynamic modelling, which, despite simpler aerodynamics compared to conventional wind turbines, presents challenges in determining relevant random variables for the SM. PCE may also be beneficial in bridge engineering, where wind loads are crucial, as these structures generally have simpler aerodynamics.

- The application of the POD-PCE methodology focused on normal power production conditions (DLC 1.2). One challenge encountered was energy distribution across considerable POD modes in unsteady wind time series. Alternatively, applying POD-PCE for gust-related DLCs, such as DLC 2.3 or DLC 4.2, may capture energy in fewer modes, facilitating the use of POD-PCE SMs.

- As turbines increase in size, the limitations of BEM become more apparent, particularly in handling the flexibility of larger wind turbine blades [4]. Lifting line models, though computationally more expensive [5], present an opportunity for SM development. The feasibility of constructing intrusive PCE SMs on lifting line models has been demonstrated [1], suggesting potential areas for further exploration.

- In Chapter 3, I employed a sequential machine learning method to construct a SM capable of mapping wind time series to DEL. The next rational step in this research is to extend this approach to offshore wind and investigate its behaviour with floating offshore wind. It is worth noting that I did not account for any yaw misalignment in the simulations, which could be a valuable direction for future research. Additionally, developing a more complex wake model and observing its effect on the SM under varying wind directions would be an exciting area of exploration. In a related area of interest, I could train a SM on the loads from one turbine in a freestream and extend it to other turbines in a

wind farm, leveraging transfer learning despite differences in the wakes for each turbine.

- During the research on sequential ML, I developed methodologies to predict load time series from wind time series utilizing SMs. However, since I primarily use load time series to extract extreme or fatigue loads, I have set that project aside. One interesting area for future work is revisiting the time-series to time-series approach and exploring the possibility of generating load time series from wind time series. Different methodologies can be employed here, but there are two particularly encouraging ones. One approach involves using two autoencoders [2], one for wind time series and one for load time series. After training these models, we connect the encoder and latent space of the wind time series to the latent space and decoder of the load time series, with a FCNN mapping between the latent spaces. The second approach is to use transformers [9]. However, transformers are known for being data-hungry [3], which may pose a challenge in this context. A time-series to time-series SM has different applications, especially for controllers and time-marching DT. Throughout my research, I did not have access to high-resolution field measurement data on real-world operating machines. Therefore, a potential avenue for future work would be to build a SM using such data. However, since high-resolution measurement data is rare, I recommend training a SM based on synthetic data. With transfer learning, one could then develop a SM based on the measurement data.

- Regarding the applications of the TCN-FCNN, a possible future work could be to use the SM model for the optimization of wind farm layout, particularly for floating wind turbines where the wake effect may have a notable impact on the structural loads [10]. This involves employing the SM to predict the loads on a floating offshore wind in the wake, which is a more complex dynamic. Although the TCN-FCNN model was not intended for the wind turbine design optimization loop, it would be interesting to explore its potential for such a purpose. One possible future work is to train the SM on a large dataset and then use trained SM for design optimization. After each optimization loop, the updated model can be used to generate a smaller database with fewer simulations, but use the TL to retrain the SM. The updated SM can then generate a large dataset that

can be used for design optimization.

- I used a publicly available wind turbine model for the AGPR study. One of my challenges was the lack of data concerning the turbine for which I had access to the SCADA data. In the future, having access to the structural dynamics characteristics and controller settings of the measurement wind turbines would be very helpful. This information would allow us to fine-tune our model and controller before running the simulations. This process would significantly improve the quality of our simulations. On another note, although the simulations were based on the power production DLC, I found that the SCADA data contained much more information than that. In future studies, one could identify the DLC based on the reported errors in SCADA and run them to cover a wider range of wind turbine behaviour. Another challenge I encountered was the lack of data for higher wind speeds. Ensuring access to a longer period of SCADA data would resolve the issue.

- The data-driven model utilized AGPR, but there are numerous other options available. One potential avenue for future work could be to adopt a deterministic approach and train an FCNN on the hybrid data. This approach involves utilizing FCNNs to map the wind speed mean and std from the SCADA to the QoI of the loads from the hybrid database. Additionally, exploring probabilistic neural networks [7] for this problem could be intriguing. One drawback of AGPR is the assumption that the output distribution is a Gaussian distribution, whereas the results in Chapter 4 indicate that it is not. Future work could include implementing a probabilistic regression model that is not confined to the Gaussian distribution [6]. Another interesting approach could be to use a transformer [9] on a lengthy stretch of SCADA data, with transformer training involving the DEL at the various turbine components. This would entail creating a hybrid database of simulations based on a lengthy stretch of SCADA data, calculating the DEL for each simulation, and training a transformer that maps the SCADA data to the DEL. Then, by providing a new SCADA, we can hope that it will accurately predict the DEL.

- The research conducted on AGPR was primarily based on simulation data as I did not have access to instrumented turbine data. However, it would be extremely beneficial to verify the simulation model and AGPR output for DEL and other QoI by utilizing measurement data. This would significantly enhance the research. Another potential future initiative is to implement this approach on an offshore wind turbine equipped with wind and wave measurement instruments and possibly an array of sensors. If such a turbine is available, it would facilitate the validation of the simulations and AGPR model.

5.4 Proposition

In the Netherlands, a component of composing a Ph.D. dissertation involves articulating propositions. These propositions ought to be both contestable and defensible, serving to demonstrate that the Ph.D. candidate has not only mastered their specific research area but also honed their capacity for philosophical thinking. Essentially, they are required to illustrate that *philosophy* remains an integral aspect of the doctoral degree. In adherence to this esteemed tradition, I present the following propositions:

- In data-driven modelling, the significance of data sampling often surpasses the choice of the model utilized.
- Addressing climate change effectively requires policy changes rather than technological advancements, achievable through democratic or non-democratic means.
- The commodification of climate change risks reducing our response to merely adopting marginally less harmful consumerism rather than addressing the fundamental issue of excessive consumerism itself.
- Pursuing unlimited development on a planet with finite resources is unsustainable.
- Providing free public mental health services is crucial for addressing the mental health crisis.[1]

[1] "The State of Mental Health in America", Mental Health America. `https://www.mhanational.org/issues/state-mental-health-america`

- High levels of individualism in North America are a key component of the widespread loneliness epidemic.[2]
- The commodification of racial, gender, and sexual minorities trivializes their struggles, reducing them to the level of commercial goods.
- Cancel culture may inadvertently perpetuate more discrimination than it aims to resolve.
- Implementing a high tax on the wealthiest 1% to fund sustainable infrastructure and farming, scientific research, free public healthcare, and education can reduce excessive wealth inequality in modern societies.[3]
- Humans often struggle with accurately extrapolating and quantifying uncertainties, leading to overly optimistic or pessimistic predictions.
- Making decisions and how we act on them are more important than the decisions themselves.
- Online dating resembles a Monte Carlo sampling of normal distributions, seeking samples a couple of standard deviations from the mean.
- "The pandemic made it clear that science touches everything, and everything touches science."[4]

[2]Office of the Surgeon General (OSG), Our Epidemic of Loneliness and Isolation: The U.S. Surgeon General's Advisory on the Healing Effects of Social Connection and Community. `http://www.ncbi.nlm.nih.gov/books/NBK595227/`

[3]"Tax on Extreme Wealth", Bernie Sanders Official Website. `https://berniesanders.com/issues/tax-extreme-wealth/`

[4]E. Yong, "What Even Counts as Science Writing Anymore?," The Atlantic. `https://www.theatlantic.com/science/archive/2021/10/how-pandemic-changed-science-writing/620271/`

Bibliography

[1] M. Fluck. *Stochastic Methods for Unsteady Aerodynamic Analysis of Wings and Wind Turbine Blades.* Thesis, 2017. URL `https://dspace.library.uvic.ca/handle/1828/7981`.

[2] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi. Escaping the Big Data Paradigm with Compact Transformers, 2022. URL `http://arxiv.org/abs/2104.05704`.

[4] S. Hauptmann, M. Bülk, L. Schön, S. Erbslöh, K. Boorsma, F. Grasso, M. Kühn, and P. W. Cheng. Comparison of the lifting-line free vortex wake method and the blade-element-momentum theory regarding the simulated loads of multi-MW wind turbines. *Journal of Physics: Conference Series*, 555(1):012050, 2014. ISSN 1742-6596. doi: 10.1088/1742-6596/555/1/012050.

[5] K. A. R. Ismail and W. M. Okita. A Comprehensive Comparative Investigation of the Lifting Line Theory and Blade Element Momentum Theory Applied to Small Wind Turbines. *Journal of Energy Resources Technology*, 144(082110), 2022. ISSN 0195-0738. doi: 10.1115/1.4053066.

[6] Y. Kındap and S. Godsill. Non-Gaussian Process Regression, 2022.

[7] B. Mohebali, A. Tahmassebi, A. Meyer-Baese, and A. H. Gandomi. Chapter 14 - Probabilistic neural networks: A brief overview of theory, implementation, and application. In P. Samui, D. Tien Bui, S. Chakraborty, and R. C. Deo, editors, *Handbook of Probabilistic Models*, pages 347–367. Butterworth-Heinemann, 2020. ISBN 978-0-12-816514-0. doi: 10.1016/B978-0-12-816514-0.00014-X.

[8] L. Pinheiro Cinelli, M. Araújo Marins, E. A. Barros da Silva, and S. Lima Netto. Variational Autoencoder. In L. P. Cinelli, M. A. Marins, E. A. Barros da Silva, and S. L. Netto, editors, *Variational Methods for Machine Learning with Applications to Deep Networks*, pages 111–149. Springer International Publishing, Cham, 2021. ISBN 978-3-030-70679-1. doi: 10.1007/978-3-030-70679-1_5.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html`.

[10] A. S. Wise and E. E. Bachynski. Wake meandering effects on floating wind turbines. *Wind Energy*, 23(5):1266–1285, 2020. ISSN 1099-1824. doi: 10.1002/we.2485.